7 U.S. PTO
09/19/00

PTO/SB/05 (2/98) (modified)
Approved for use through 9/30/2000, OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

NEW UTILITY PATENT APPLICATION TRANSMITTAL *(only for new nonprovisional applications under 37 CFR 1.53(b))*	*Attorney Docket Number*	4483 US
	First Named Inventor	Pavlovic
	Total Pages in this Submission	72
	Express Mail Label No.	EL482472084US

APPLICATION ELEMENTS

1. ☒ Fee Transmittal Form (in duplicate)
   ☒ Check Enclosed $345.00
2. ☒ Specification (21 pages)
   (preferred arrangement set forth below)
   - Descriptive Title of the Invention
   - Cross Reference(s) to Related Case(s)
   - Statement Regarding Fed sponsored R & D
   - Background of the Invention
   - Brief Summary of the Invention
   - Brief Description of the Drawing(s)
   - Detailed Description
   - Claim or Claims
   - Abstract of the Disclosure
3. ☒ Drawing(s) *(when necessary per 35 USC 113)* (44 pages)
4. Oath or Declaration
   a. ☒ New Declaration (3 pages)
      ☐ Executed
   b. ☐ Copy from a prior application (37 CFR 1.63(d))
      (for continuation/divisional with Box 17 completed)
      i. ☐ DELETION OF INVENTOR(S)
         Signed statement attached deleting inventor(s) named in the prior application, see 37 CFR 1.63(d)(2) and 1.33(b).
5. ☐ Incorporation by Reference *(useable if Box 4b is checked)*. The entire disclosure of the prior application, from which a copy of the oath or declaration is supplied under Box 4b, is considered as being part of the disclosure of the accompanying application and is hereby incorporated by reference therein.

ACCOMPANYING APPLICATION PARTS

6. ☐ Assignment & Assignment Recordation Cover Sheet
7. ☐ Certified Copy of Priority Document(s)
   (if foreign priority is claimed)
8. ☐ Information Disclosure Statement & PTO-1449
   ☐ Copies of IDS Citation(s)
9. ☐ Preliminary Amendment
10. Small Entity Statement
    ☒ New Statement enclosed (1 page)
    ☐ Statement filed in prior application. Status still proper and desired
11. ☒ Return Postcard
12. ☐ ______
13. ☐ ______
14. ☐ ______
15. ☐ ______
16. ☐ ______

1c515 U.S. PTO
09/665179
09/19/00

ADDRESS TO:

Box Patent Application
Commissioner for Patents
Washington, D.C. 20231

17. **If a CONTINUING APPLICATION,** *check appropriate box and supply the requisite information below and in a preliminary amendment:*
☐ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) of prior application No: ___/_______
Prior application information: *Examiner:* ____________ *Group/Art Unit:* ________

18. CORRESPONDENCE ADDRESS

NAME	Laura A. Majerus Fenwick & West LLP				
ADDRESS	Two Palo Alto Square				
CITY	Palo Alto	*STATE*	CA	*ZIP CODE*	94306
COUNTRY	U.S.A.	*TELEPHONE*	(650) 858-7152	*FAX*	(650) 494-1417
Name (Print/Type)	Laura A. Majerus	*Registration No. (Attorney/Agent)*	33,417		
Signature	Laura Majerus	*Date*	September 19, 2000		

PTO/SB/17 (6-95)(modified)
Approved for use through 11/30/96, OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

0002/PTO(modified) Rev. 10/95 — U.S. Department of Commerce, Patent and Trademark Office

FEE TRANSMITTAL

TOTAL AMOUNT OF PAYMENT

Subtotal (1) + Subtotal (2) + Subtotal (3) = ($)345.00

Complete if Known	
Application Number	To Be Assigned
Filing Date	Herewith
First Named Inventor	Pavlovic
Group Art Unit	To Be Assigned
Examiner Name	To Be Assigned
Attorney Docket Number	4483 US

METHOD OF PAYMENT

1. The Commissioner is hereby authorized to:

☐ Charge the indicated fees to the below mentioned deposit account.

☒ Charge any additional fee required under 37 CFR 1.16 - 1.21 or credit any over payments to the below mentioned deposit account. †

☐ Charge the Issue Fee set in 37 CFR 1.18 at the Mailing of the Notice of Allowance, 37 CFR 1.311(b) to the below mentioned deposit account.

Deposit Account Number: 19-2555
Deposit Account Name: FENWICK & WEST LLP

A Duplicate Copy of this authorization is attached

2. ☒ **Payment Enclosed:**
☒ Check ☐ Other

FEE CALCULATION (fees effective 11/12/98)

1. FILING FEE

Large Entity Fee Code/Fee	Small Entity Fee Code/Fee	Fee Description	Fee Due
101/$690	201/$345	Utility Filing	345
106/$310	206/$155	Design Filing	
108/$690	208/$345	Reissue	
114/$150	214/$75	Provisional Filing	

SUBTOTAL (1) ($) 345.00

2. CLAIMS

Large Entity Fee Code/Fee	Small Entity Fee Code/Fee	Fee Description
103/$18	203/$9	Claims in excess of 20
102/$78	202/$39	Independent claims in excess of 3
104/$260	204/$130	Multiple dependent claim
109/$78	209/$39	Reissue independent claims over original patent
110/$18	210/$9	Reissue claims in excess of 20 and over original patent

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code/Fee	Small Entity Fee Code/Fee	Fee Description	Fee Due
105/$130	205/$65	Surcharge - late filing fee or oath	
127/$50	227/$25	Surcharge-late provisional filing fee or cover sheet	
147/$2,520	147/$2,520	For filing a request for reexamination	
115/$110	215/$55	Extension for response within first month†	
116/$380	216/$190	Extension for response within second month†	
117/$870	217/$435	Extension for response within third month†	
118/$1,360	218/$680	Extension for response within fourth month†	
128/$1,850	228/$925	Extension for response within fifth month†	
119/$300	219/$150	Notice of Appeal	
141/$1,210	241/$605	Petition to revive unintentionally abandoned application	
142/$1,210	242/$605	Utility Issue Fee (Or Reissue)	
143/$430	243/$215	Design Issue Fee	
122/$130	122/$130	Petitions to the Commissioner	
123/$50	123/$50	Petitions related to provisional applications	
126/$240	126/$240	Submission of Information Disclosure Statement	
581/$40	581/$40	Recording each patent assignment per property (times number of properties)	
146/$690	246/$345	Filing a submission after final rejection (37 CFR 1.129(a))	
149/$690	249/$345	For each additional invention to be examined (37 CFR 1.129(b))	
		Other fee *(specify)*:	
		Other fee *(specify)*:	

SUBTOTAL (3) ($) 0.00

For	(Col. 1) No. of Existing Claims		(Col. 2) Highest No. Previously Paid For		(Col. 3) Extra**		Fee		Fee Due
TOTAL		minus*	20 or 0	=		x		=	
INDEP		minus*	3 or 0	=		x		=	
[ ] First presentation of multiple dependent claim								=	

* *Subtract the greater number of Col. 2*
** *If the difference between Col. 1 and Col. 2 is less than zero, then enter "0" in Col. 3*

SUBTOTAL (2) ($) 0.00

SUBMITTED BY		Complete (if applicable)	
Typed or Printed Name	Laura A. Majerus	Reg. Number	33,417
Signature	Laura Majerus	Date	September 19, 2000

† *Request for Extension of Time per 37 CFR 1.136 (a)(3) made hereby*
Rev. 11/04/99

11128/04483/DOCS/1093759.1

PTO/SB/11 (6-95) (modified)
Approved for use through 07/31/96 OMB 0651-0031
Patent and Trademark Office U.S. DEPARTMENT OF COMMERCE

VERIFIED STATEMENT CLAIMING SMALL ENTITY STATUS (37 CFR 1.9(e) & 1.27(d))-NONPROFIT ORGANIZATION

Docket Number (Optional):

Applicant or Patentee: Dusko Pavlovic, Douglas Smith and Junbo Liu

Application or Patent No. To Be Assigned

Filing Date or Issue Date: Herewith

Title: Method and Apparatus for Determining Colimits of Hereditary Diagrams

I hereby declare that I am an official empowered to act on behalf of the nonprofit organization identified below

NAME OF NONPROFIT ORGANIZATION Kestrel Institute

ADDRESS OF NONPROFIT ORGANIZATION 3260 Hillview Avenue

Palo Alto, California 94304

TYPE OF NONPROFIT ORGANIZATION:

[ ] UNIVERSITY OR OTHER INSTITUTION OF HIGHER EDUCATION

[X] TAX EXEMPT UNDER INTERNAL REVENUE SERVICE CODE (26 U.S.C. 501(a) and 501(c)(3))

[ ] NONPROFIT SCIENTIFIC OR EDUCATIONAL UNDER STATUTE OF STATE OF THE UNITED STATES OF AMERICA
(NAME OF STATE ______)
(CITATION OF STATUTE ______)

[ ] WOULD QUALIFY AS TAX EXEMPT UNDER INTERNAL REVENUE SERVICE CODE (26 U.S.C. 501(a) and 501(c)(3)) IF LOCATED IN THE UNITED STATES OF AMERICA

[ ] WOULD QUALIFY AS NONPROFIT SCIENTIFIC OR EDUCATIONAL UNDER STATUTE OF STATE OF THE UNITED STATES OF AMERICA IF LOCATED IN THE UNITED STATES OF AMERICA
(NAME OF STATE ______)
(CITATION OF STATUTE ______)

I hereby declare that the nonprofit organization identified above qualifies as a nonprofit organization as defined in 37 CFR 1.9(e) for purposes of paying reduced fees to the United States Patent and Trademark Office regarding the invention described in:

[X] the specification filed herewith with title as listed above
[ ] the application identified above.
[ ] the patent identified above.

I hereby declare that rights under contract or law have been conveyed to and remain with the nonprofit organization regarding the above identified invention. If the rights held by the nonprofit organization are not exclusive, each individual, concern or organization having rights in the invention must file separate verified statements averring to their status as small entities and that no rights to the invention are held by any person, other than the inventor, who would not qualify as an independent inventor under 37 CFR 1.9(c) if that person made the invention, or by any concern which would not qualify as a small business concern under 37 CFR 1.9(d) or a nonprofit organization under 37 CFR 1.9(e).

Each such person, concern or organization having any rights in the invention is listed below:

[X] no such person, concern, or organization exists.
[ ] each such person, concern or organization is listed below

I acknowledge the duty to file, in this application or patent, notification of any change in status resulting in loss of entitlement to small entity status prior to paying, or at the time of paying, the earliest of the issue fee or any maintenance fee due after the date on which status as a small entity is no longer appropriate. (37 CFR 1.28(b))

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application, any patent issuing thereon, or any patent to which this verified statement is directed.

NAME OF PERSON SIGNING Dr. Cordell Green

TITLE IN ORGANIZATION OF PERSON SIGNING Director, Kestrel Institute

ADDRESS OF PERSON SIGNING 3260 Hillview Avenue, Palo Alto, CA 94304

SIGNATURE Cordell Green DATE 9/18/00

11128/0448
3/DOCS/10
93465 1

U.S. PATENT APPLICATION

for

METHOD AND APPARATUS FOR DETERMINING COLIMITS OF HEREDITARY DIAGRAMS

Inventors:
Dusko Pavlovic
Douglas R. Smith
Junbo Liu

Fenwick & West LLP
2 Palo Alto Square
Palo Alto, A 94306
650-494-0600

Express Mail Number EL482472084US

METHOD AND APPARATUS FOR DETERMINING COLIMITS OF HEREDITARY DIAGRAMS

Inventors:

Dusko Pavlovic

Douglas R. Smith

Junbo Liu

Related Applications:

This application claims priority under 35 U.S.C. § 119(e) to U.S. Application Serial No. 60/155,271 entitled "Method and Apparatus for Determining Colimits of Hereditary Diagrams" of Pavlovic et al., filed September 19, 1999, which is herein incorporated by reference in its entirety.

BACKGROUND OF THE INVENTION

The present invention relates generally to system design and, specifically, to a method and system used to refine stratified design specifications, presented as hereditary diagrams.

The design of systems, such as computer systems or engineering systems is a complex process. While it is possible to design systems from scratch using a minimum of design tools, most modern designers use tools to represent and manipulate designs for complex systems.

SUMMARY OF THE INVENTION

In the described embodiment of the present invention, a user specifies his design using a specification language. Specification software manipulates the specified design to yield a more detailed system design. Some of these manipulations involve use of a library of specifications.

Specifications are the primary objects in the described specification language. A specification can represent any system or realm of knowledge such as computer programming or circuit design and describes a concept to some degree of detail. To add properties and extend definitions, the described specification software allows the user to create new specifications that import or combine earlier specifications. This process is called refinement. Composition and refinement are the basic techniques of application development in the described specification software. A user composes simpler specifications into more complex ones, and refines more abstract specifications into more concrete ones. Refining a specification creates a more specific case of it.

In the described embodiment, specifications can represent an object or concept. A complex specification can be presented as a diagram of simpler specifications. A software specification is a formal representation of objects or concepts that come about in a software development project. In the described embodiment, a complex specification can be composed and refined as a diagram of simpler specifications; still more involved specifications can be composed as diagrams of such diagrams; and so on. Large specifications are thus subdivided into diagrams of smaller specifications. The process of software design is stratified into such diagrams, diagrams of diagrams and so on. This is what is meant by the expression "hereditary diagrams of specification." A diagram includes:

- A set of nodes (or vertices)
- A set of arcs (or edges or arrows), and
- Two mappings, assigning two nodes to each arc: its source-node and its target-node.

The nodes of a diagram of specifications are formal specifications, capturing the relevant objects and concepts to be specified, the arcs of a diagram of specifications are the "specification morphisms," capturing the relationships between the nodes: how some specifications inherit or share the structure specified in others. Diagrams thus provide a

graphically based method for software development and refinement, allowing "modular decomposition" and reuse of software specifications.

The described embodiments of the software development tool support:

- Specification refinement: deriving a more concrete specification from a more abstract specification by adding more structural detail
- Code generation: when enough structural detail has been specified to determine concrete programming structures suitable to perform the required task, code in a suitable language is generated.
- Colimit determination

In general, determination of a colimit is a destructive operation, resulting in the loos of information about the involved diagrams. The described embodiments of the invention protect and retain the diagrams by folding them into a node. Since the described embodiment allow for diagrams of diagrams, this protection can occur in a multi-level diagram of diagrams.

Nodes of a diagram show the objects or concepts and arcs between the nodes show relationships (morphisms) between the nodes. Diagrams are used primarily to create sets of objects and to specify their shared parts, so that the individual parts can be combined. Specifications can also be defined to be hereditary diagrams.

The described specification software allows a user to derive a more concrete specification from a more abstract specification. In general, the complexity of a specification is increased by adding more structural detail. The following techniques are preferably used (separately or together) to refine specifications:

-the import operation, which allows a user to include earlier specifications into a later one;

-the translate operation, which allows a user to rename the parts of a specification; and

-the colimit operation, which glues concepts together into a shared union along shared sub-concepts.

Use of diagrams (and hereditary diagrams) allows the user to retain information about a specification during the design process. The described embodiment of the present invention allows a user to define a specification that is a hereditary diagram and to perform the colimit operation on the hereditary diagram.

The described embodiments include specification diagrams and compute co-limits in this category. Furthermore, the described embodiments iterate this procedure, yielding the category of hierarchical diagrams, and computes colimits for these hierarchal diagrams.

Brief Description of the Drawings

Fig. 1 is a block diagram of the overall architecture of an embodiment of the present invention.

Figs. 2(a) and 2(b) are flow charts showing step-wise refinements of a specification.

Figs. 3(a) and 3(b) show a conceptual example of a colimit operation.

Figs. 4(a) and 4(b) show another conceptual example of a colimit operation.

Fig. 5 shows an example of the colimit operation for a specification.

Fig. 6 shows an example of the colimit operation for a hereditary diagram.

Fig. 7 shows another example of the colimit operation for a hereditary diagram.

Figs. 8(a), 8(b), and 8(c) show an example user interface for the colimit operation of a hereditary diagram.

Figs. 9(a)-9(j) show an example of operations initiated by the user to further illustrate the colimit operation for a hereditary diagram

Fig. 10 is a flow chart of a method performed by the exemplary specification software to determine a colimit of a hereditary diagram.

Fig. 11 is a flow chart showing a first part of the method of Fig. 10 to determine a diagram of shape categories.

Figs. 12(a) and 12(b) provide an example of a hereditary diagram.

Fig. 13 provides a more detailed example of the hereditary diagram of Fig. 12.

Fig. 14 provides an example of a diagram of shape categories for the hereditary diagram of Fig. 13.

Figs. 15(a)-15(d) provide a more detailed example of the diagram of Fig. 14.

Fig. 16 is a flow chart showing additional portions of a first part of the method of Fig. 10 to determine a colimit of a diagram of shape categories.

Fig. 17 provides an example of a colimit of a diagram of shape categories.

Figs. 18(a)-18(f) provide a more detailed example of the colimit of Fig. 17.

Fig. 19 is a flow chart showing a second part of the method of Fig. 10 to extend a diagram in the hereditary diagram in accordance with the colimit of the shape diagram.

Fig. 20 provides examples of extended diagrams.

Figs. 21(a)-21(f) provide a more detailed example of an extended diagram.

Fig. 22 is a flow chart showing a third part of the method of Fig. 10.

Fig. 23 provides an example of a taking a colimit of extended diagrams.

Figs. 24-26 provide a more detailed example of taking a colimit of extended diagrams to yield a colimit of the original hereditary diagram.

Fig. 27 shows example data structures used in a preferred embodiment.

Fig. 28 shows example data structures used in a preferred embodiment.

Fig. 29 is a diagram showing a conceptual view of a set of arcs and a set of nodes, with a target mapping and a source mapping between them.

DETAILED DESCRIPTION

General Discussion

The described embodiment provides a software tool for building, manipulating, and reusing a collection of related specifications. The tool allows a user to describe concepts in a formal language with rules of deduction. It includes a database (library) that stores and manipulates collections of concepts, facts, and relationships. The present invention can be used to produce more highly refined specifications until a concrete

level of abstraction is reached. For example, a specification can be refined until it reaches the computer source code level. As another example, a specification can be refined until it reaches the circuit level.

Referring now to Fig. 1, there is shown a block diagram of the overall architecture of an embodiment of the present invention. Fig. 1 includes a data processing system 100 including a processor 102 and a memory 104. Memory 104 includes specification software 110, which implements the refinement methods defined herein. Specification software 110 preferably implements a graphical user interface (GUI) that allows a user to define specifications and morphisms and that allows a user to indicate refinements to be performed on the specifications. Specification software 110 includes or accesses a database 112 that includes definitions of specifications and diagrams. The specification being refined is stored in memory 114. The refinement operations indicated by the user can result in computer code 116 if the user chooses to perform refinements to the computer code level.

Figs. 2(a) and 2(b) are flow charts showing step-wise refinements of a specification during an exemplary design process. In element 202 of Fig. 2(a), the user is allowed to define/enter software specifications, diagrams, and hereditary diagrams (also called a "hierarchical diagram" or a "diagrams of diagrams"). Specifications are the primary objects defined by a user. In the described embodiment, specifications can represent a simple object or concept. A specification can also be a diagram, which is a collection of related objects or concepts. As shown in Figure 29, nodes of a diagram show the objects or concepts and arcs between the nodes show relationships (morphisms) between the nodes. Diagrams are used primarily to create sets of objects and to specify their shared parts, so that the individual parts can be combined. Specifications can also be defined to be hereditary diagrams, where at least one object in a node of the diagram is another diagram.

Specifications can be defined in any appropriate specification language, such as the SLANG language defined by the Kestrel Institute of Palo Alto, CA. SLANG is

defined in the SLANG Users Manual, available from the Kestrel Institute of Palo Alto, CA. The Slang Users Manual is herein incorporated by reference. A specification can represent any system or realm of knowledge such as computer programming or circuit design and describes a concept to some degree of detail.

In element 204, the user is allowed to start refining his specifications, diagrams, and hereditary diagrams. To add properties and extend definitions, the described specification software allows the user to create new specifications that import or combine earlier specifications. This process is called refinement. Composition and refinement are the basic techniques of application in the described specification software. A user composes simpler specifications into more complex ones, and refines more abstract specifications into more concrete ones. Refining a specification creates a more specific case of it.

The described specification software allows a user to derive a more concrete specification from a more abstract specification. In general, the complexity of a specification is increased by adding more structural detail. The following techniques, among others, are preferably used (separately or together) to refine specifications:

-the import operation, which allows a user to include earlier specifications into a later one;

-the translate operation, which allows a user to rename the parts of a specification; and

-the colimit operation, which glues concepts together into a shared union along shared sub-concepts.

Fig. 2(b) is a flow chart of a method for refining a specification. The user indicates a refinement operation, which is then performed by specification software 110. Fig. 2(b) shows three examples of refinement operations. It will be understood that other refinements are possible. In element 216, the user indicates that a spec or diagram is to be imported. In element 218, the user indicates finding a colimit of a hereditary diagram. In element 220, the user indicates a translation of a spec or diagram.

In element 206 of Fig. 2(a), the user refines his specification to a level of generating computer code. A user may choose not to refine a specification to this level. The refinement process can be used for purposes other than generating computer source code. For example, the refinement process can be used to help understand a specification. As another example, the refinement process can be used to help verify the consistency of a specification.

The Colimit Operation

Figs. 3(a) and 3(b) show a conceptual example of a colimit operation. A colimit is also called "composition" or a "shared union." A "pushout" is a colimit in which a colimit is taken of a parent node and its two children nodes. It will be understood that the examples of Figs. 3 and 4 are somewhat simplified and are provided to aid in understanding of the colimit operation. In Fig. 3, the user has defined a spec "car" 302. This specification 302 has been refined by the user as red car 304 and fast car 306. Thus, the arcs from node 302 to 304 and 302 to 306 are labeled with an "i" (for instantiation/import). In Fig. 3(a), the "defining diagram" shows only the spec/morphism diagram from which the colimit is formed. Fig. 3(b) shows a "cocone diagram," which also shows the colimit and the cocone morphisms (labeled "c").

In the described embodiment, the GUI labels arcs as follows, although any appropriate labeling and morphisms could be used (or none).

i: instantiation morphism

d: definitional translation

t: transitional morphsim

c: cocone morphism

id: identity morphism

The defining diagram for a colimit is not limited to a three node diagram. A colimit can be taken of any diagram. An example of a different diagram shape is shown in Fig. 3(b). In the colimit operation, any type of node related by morphisms in the

diagrams are mapped to the same type of node in the colimit. Conversely, any unrelated types are mapped to different types in the colimit. The same is true of operations.

When you compose specifications, types or operations that have the same names in different component specifications might be mapped to different result operations. For example, suppose spec A and spec B are combined to form spec C. Both A and B have operations named concat, but the operations do not work the same way, and need to be differentiated in spec C. In this case, specification software 110 generates unambiguous names in the colimit. Similarly, types and operations that have different names in the component specifications can be mapped to a single element in the colimit. For example, the operation concat in spec A and add in spec B might both be mapped to a single concatenation operation in the colimit spec C. In this case, the resulting element preferably has both names.

Fig. 5 shows a more realistic example of the colimit operation for a specification. In this example, a virtual memory (VM) is a parameter of the operating system (OS). Suppose we want to formally specify a simple operating system (OS). There are large fragments of the theory that can be abstracted away. In other words, the structure of the system does not depend on a particular virtual memory (VM) implementation. Thus, the formal VM requirements can be taken as a parameter of the formal OS specification. Similarly, a particular VM system, VM_0, can be a parametric in paging policies (PP). Thus, the parameter VM can be instantiated to another parametric specification VM_0.

In this way, a complex system naturally decomposes into simpler components that can be refined independently. When all components are implemented, an implementation of the whole can be automatically generated: an operating system with a particular virtual memory management and with a particular paging policy.

Use of diagrams (specifically, hereditary diagrams) allows the user to retain information about a specification during the design process. Taking the colimit of simple specifications can destroy the structure of the spec. The described embodiment

of the present invention allows a user to define a specification that is a hereditary diagram and to perform the colimit operation on the hereditary diagram. This carrying information in a diagram brings the colimit operation into lazy mode. Fig. 6 shows an example of the colimit operation for a hereditary diagram. Various intermediary choices can be made by the user as to how to define a diagram. For example, one may wish to instantiate the virtual memory parameter VM to VM_0, but to keep the page-in policy parameter PP open. The pspec VM_0 can then be protected as a diagram 650. The colimit operation can then be applied in the category of diagrams, rather than specs. Note that Fig. 6 shows an example of a hereditary diagram in which at least one node is a diagram.

The parameter VM to be instantiated for, lifts to a trivial diagram as well as the spec OS. The colimit of the resulting diagram yields the spec OS parametric over PP as a diagram.

Fig. 7 shows another example of the colimit operation for a hereditary diagram. Implementation details of colimits of hereditary diagrams are discussed below in connection with Figs. 10-27. Shape changes of even simple diagrams quickly become too complex for human beings to solve intuitively. An automated method is needed, such as that shown in detail herein.

Figs. 8(a), 8(b), and 8(c) show an example graphical user interface (GUI) for the colimit operation of a hereditary diagram. The display of Figs. 8 and 9 preferably are generated by specification software 110. In Fig. 8(a), the user has defined a hereditary diagram. An initial (parent) spec is named Bag-Diagram. Fig. 9(c) shows details of Bag-Diagram. (The user may or may not choose to display the detail of the diagram Bag-Diagram and may instead display only the name of the diagram as shown in Fig. 8(a)). In this example, the user has refined the parent spec twice, to yield: Bag-as-Seq-Dg and Bag-Seq-over-Linear-Order. Figs. 9(d) and 9(e) show details of these diagrams. (The user may or may not choose to display the detail of the diagrams and may instead display only the names of the diagrams as shown in Fig. 8(a)).

In Fig. 8(b), the user has selected the diagram having Bag-Diagram as its parent node and has indicated that he wishes to refine the hereditary diagram spec via the colimit operation. Although the disclosed interface uses a drop-down menu to allow the user to indicate the colimit operation, any appropriate interface can be used. In Fig. 8(c), the colimit is named Diagram-5. Fig. 9(j) shows details of this diagram. (The user may or may not choose to display the detail of the diagram and may instead display only the name of the colimit diagram as shown in Fig. 8(c)).

Figs. 9(a)-9(j) show an example of operations initiated by the user to further illustrate the colimit operation for a hereditary diagram. Fig. 9(a) shows an initial hereditary diagram. Fig. 9(b) shows an example of the result of the colimit operation indicated by the user. Fig. 9(c) shows an expansion of the Bag-Diagram requested by the user. Fig. 9(d) shows an expansion of the Bag-as-Sequence-Diagram requested by the user. Fig. 9(e) shows an expansion of the Bag-Seq-over-Linear-Order-Diagram requested by the user.

Figs. 9(f)-9(i) show details of determination of the colimit of the hereditary diagram of Fig. 9(a). Fig. 9(f) shows a shape of the shape colimit, which is the shape that the colimit will eventually have. Fig. 9(g) shows an extension of the Bag-Diagram in accordance with the shape of the colimit. Fig. 9(h) shows an extension of the Bag-as-Sequence-Diagram in accordance with the shape of the colimit. Fig. 9(i) shows an extension of the Bag-Seq-over-Linear-Order-Diagram in accordance with the shape of the colimit. Fig. 9(j) shows an expanded version of Diagram-5, which is the colimit of the hereditary diagram. Note that the colimit has the shape of the diagram of Fig. 9(f).

A Preferred Implementation of the Colimit Operation for Hereditary Diagrams

Fig. 10 is a flow chart of a method preferably performed by the specification software 110 to determine a colimit of a hereditary diagram. In element 1002, the user indicates that he wants to take the colimit of a hereditary diagram. An example of a GUI to accomplish this indication is shown in Fig. 8(b). When software 110 receives such an

indication from the user, software 110 preferably performs the remaining elements of Fig. 10. In element 1004, software 110 extracts the shapes of the nodes and functors of the hereditary diagram to yield a diagram of shape categories. Details of this element are shown in Fig. 11. Once a diagram of shape categories has been determined, software 110 determines, in element 1004, a colimit of the diagram of shape categories, as discussed below in connection with Fig. 16.

Software 110 then in element 1006 determines an extension of each diagram in the hereditary diagram in accordance with the shape colimit, as is discussed below in connection with Fig. 19. Extending each diagram in the hereditary diagram brings all diagrams in the hereditary diagram to the same shape, so that it is possible to take the pointwise colimit of the extended diagram. Then software 110 in element 1008 determines the colimit of the hereditary diagram using the extended diagrams, as discussed below in connection with Fig. 22. Once the colimit is determined, it can be stored in memory, saved, or displayed, as the user decides.

The following discussion of a preferred software program uses certain abstract concepts, which are presented in Table 1, which forms a part of this specification.

I. Determining a Shape Colimit

Fig. 11 is a flow chart showing a first part of the method of Fig. 10 to determine a diagram of shape categories. The elements of Fig. 11 are performed for each diagram in the hereditary diagram.

First, hereditary diagrams will be discussed. Fig. 12(a) shows a high-level example of a hereditary diagram having three nodes d1, d2, d3 (each of which is a diagram) and two arcs a1 and a2 (also called "arrows" or "edges). Each of arcs a1 and a2 represents a shape morphism between a pair of diagrams in the hereditary diagram. Fig. 12(b) shows a representation of a shape morphism from d1 to d2, where F is a shape functor and e is a natural transformation from d1 to (d2 composed with F). D1 and D2 are shape categories of respective diagrams d1 and d2 and SPEC is the category SPEC.

In the described embodiment, the user must specify shape morphisms between diagrams when the diagrams are created. Alternative embodiments determine this mapping heuristically by, for example, counting the number of arcs in and out of nodes or by looking at the type of the nodes.

Fig. 13 provides a more detailed example of the hereditary diagram of Fig. 12. Here, d1 is defined as having three nodes (i, ii, and iii) and two arcs (B and C). Diagram d2 is defined as having five nodes (0, 1, 2, 3, and 4) and four arcs (a, b, c, and d). Diagram d3 is defined as having two nodes (IV and V) and two arcs (A and B) .

As shown in element 104 of Fig. 10, software 110 determines diagrams of shape categories. Fig. 14 provides an example of a diagram of shape categories for the hereditary diagram of Fig. 13. Diagrams D1, D2, and D3 represent the respective shapes of d1, d2, and d3. Functions F_e and F_n provide a mapping between the arcs (edges) and nodes of each pair of shape diagrams.

Figs. 15(a)-15(d) provide a more detailed example of the diagram of Fig. 14. Mapping F1 between D1 and D2 is shown in Fig. 15(b). Mapping between D1 and D3 is shown in Fig. 15(c). Source nodes and targets nodes for arcs of the hereditary diagram are shown in Fig. 15(d).

Fig. 16 is a flow chart showing additional portions of a first part of the method of Fig. 10 to determine a colimit of a diagram of shape categories. A first element computes a colimit of the sets of nodes in the hereditary diagram. To do this, the software performs the following: Store in memory a disjoint union of all nodes (ignore arcs). Determine the equivalence relations identifying those nodes that are connected by some arc of the hereditary diagram. All nodes of the diagrams that fall in the same equivalence class are identified as a single node in the colimit. A similar element is performed to determine the colimit of the arcs. The software then considers the relationships between the equivalence classes of arcs and of nodes. For each arc in the colimit, the universal property (of the sets of arcs) determines a source node and a target node in the colimit.

Fig. 17 provides an example of a colimit D4 of a diagram of shape categories. As can be seen, several of the nodes of the colimit are created by combining nodes in the diagrams. Figs. 18(a)-18(f) provide a more detailed example of the colimit of Fig. 17. The combined nodes have been determined to belong to the same equivalence class. For example, nodes 0 or 4 of diagram D3 do not map to any nodes of D1 or D3. Thus, these nodes are not grouped. Arcs b, B, *A* and arcs c, C, *B* are grouped, as are nodes 1,3, i, iii, IV 2, ii, V. As shown in Fig. 13, the grouped nodes and arcs belong to the same equivalence classes.

Figs. 18(d)-18(f) show details of the mappings F3, F4, and F5 between the arcs and nodes or the diagrams in the hereditary diagram and D4.

II. <u>Extending the Diagrams in Accordance with the Shape Colimit</u>

In the previous section, we showed how to compute the shape of the colimit diagram. In this step we describe the method for computing the extension of a diagram along a shape morphism. The following paragraphs provide a short overview.

Let m:S -> A be a diagram and let T be the shape of the desired diagram, where f:S->T is a shape morphism. The Extension method computes a diagram E(m,f):T -> A and a natural transformations eps: m -> f;n, with the universal property that for any k:T->A and natural transformation alph:m-> f;k, there is a unique natural transformation sig:n->k such that alph factors through eps: alph = eps;(sig f) where ";" is used for vertical composition of natural transformations.

The method for computing the extension of diagram m along shape morphism f, denoted E(m,f), is.

(1) For each node t in T, we form its image under E(m,f) as follows:
form the shape f/t whose nodes are
{<s, i> | i: f(s) -> t is a path in T}
and whose arcs are

{ e | e:s->s' in S and f(e):f(s) -> f(s') in B and <s,i> and <s',i'> are nodes}. E(m,f)(t) is the (spec) colimit of the image of f/t under m.

(2) for each arc h:y->z in T, E(m,f)(h) is a unique morphism that witnesses the universality of the construction of E(m,f)(y). That is, h induces a functor from f/y to f/z and thus a diagram morphism dm: m(f/y) -> m(f/z). The composition of dm with the cocone morphisms from m(f/z) to E(m,f)(z) forms a cocone on m(f/y), so E(m,f)(h) is the unique arrow from E(m,f)(y) to E(m,f)(z) that factors the cocone arrows.

We have the following property of E:

Theorem: When T is acyclic, then colimit(m) is isomorphic to colimit(E(m,f)).

This theorem asserts that we are neither gaining nor losing information in computing the extension of a diagram along a shape morphism. We are simply changing its shape.

The colimit of diagrams enables the automated application of design theories to requirement specifications. Knowledge about various kinds of software design knowledge (such as algorithm design principles, datatype refinements, software architectures, and program optimization techniques), and other forms of design knowledge, may be represented by refinement morphisms from a diagram of the general abstract structure A required for applicability of the design knowledge, and an abstract specification diagram of a design artifact B. Such a

refinement morphism m:A->B is applied to a requirement diagram R by first constructing a "classification" morphism c:A->R, and then computing the (diagram) colimit R'. The cocone morphism cm:R->R' is a refinement of R that embodies the design knowledge in m.

Fig. 19 is a flow chart showing a second part of the method of Fig. 10 to extend a diagram in the hereditary diagram in accordance with the colimit of the shape diagram. The elements of Fig. 19 are performed for each diagram in the hereditary diagram so that each diagram yields an extended diagram. To extend one diagram, the following acts are performed. For each node in colimit D4, determine a node in the extended diagram. Thus, the extended diagram will have the same number of nodes as the shape colimit D4. The following are performed for each node n in D4: Find the nodes s in the shape diagram (for example D1) that have a path i to the node n. This yields a set of pairs of nodes s and paths i: {<s,i> , where I is a path from F(s) to node n in D4). F(s) is the node in D4 that corresponds to node(s) in D1.

For each two pairs in the set (for example, for <s_i,i> and <s_j,j>, find a path e between nodes si and sj. Because s_i and s_j have been determined to point to the same node, such a path must exist. Once each path e has been found between each pair in the set, make a graph of all the <s,i> pairs and take the colimit of the graph. This colimit forms one node in the extended diagram. Each arc in the extended diagram is uniquely defined and determined using the universality of the colimits for the nodes in the extended diagram.

Fig. 20 provides examples of three extended diagrams. The notation, for example, Lf5d1 represents the extension of diagram D1.

Figs. 21(a)-21(f) provide a more detailed example of an extended diagram. Fig. 21(a) shows details of D1 and D4 and of the mapping F5, the mapping Lf5d1 and of the mapping between D1 and the extended diagram. The other extended diagrams are determined similarly. Fig. 21(b) shows an example of finding the nodes of the extended diagram. In Fig. 21(b), the method of Fig. 19 is performed for each node in D4, yielding

a colimit that is the extension of the diagram D1. Fig. 21(c) shows an example of determining the arcs of the extension for two of the arcs of D4. In the example, arcs d and (Cc*B)* are similar, due to the example chosen.

Fig. 21(d) shows examples of specs/nodes and arcs that have been predefined by the user. These specs are used is the mapping between D1 and SPEC in Fig. 21(a). Similarly, the arcs are the arcs used in the same mapping.

Figs. 21(e) and 21(f) are examples of using the specs of Fig. 21(d) to compose several specs that will form the nodes of the extended diagram. Note that in Fig. 12(a), the nodes and arcs of the extension are defined as mapping Lf5d1. Fig. 21(e) implements the compositions of this mapping.

III. <u>Determining a Pointwise Colimit of the Extended Diagrams</u>

Fig. 22 is a flow chart showing a beginning of a third part of the method of Fig. 10.

Fig. 23 provides an example of the extended diagrams 2300, 2302, 2304, which will be used to determine the colimit of the extended diagrams.

Figs. 24-26 provide a more detailed example of taking a pointwise colimit of extended diagrams to yield a colimit of the original hereditary diagram.

Fig. 27 shows example data structures used in a preferred embodiment. Fig. 28 shows a conceptual representation of object-oriented data structures. As can be seen from the diagram, a hereditary diagram 2802 includes a plurality of objects, at least one of which 2804 is itself a diagram 2806. The details of diagram 2806 are not shown for the sake of clarity. Of course, diagram 2806 could also be a hereditary diagram. It will be understood that, although the present implementation uses an object oriented programming language, any appropriate implementation method can be used.

From the above description, it will be apparent that the invention disclosed herein provides a novel and advantageous system and method of determining colimits of hereditary diagrams.

What is Claimed is:

1. A computer-implemented method of automated software specification, comprising:

storing specification modules, with their relations displayed on a computer screen in terms of their specification morphisms, where the specification morphisms translate the specification signatures while preserving the logical structure of the specification;

determining and displaying, in response to a user command, multiple specification diagrams, each of which captures the relation between two or more specification modules, along with its specification morphisms;

building and displaying, in response to a user command, a diagram of the specification diagrams, the diagram of specification diagrams retaining the diagram morphisms of the specification diagrams; and

computing the colimits of the hereditary diagram of diagrams to compose large software modules while preserving the decomposition of the involved components.

2. A computer-implemented method for determining a colimit of a hereditary diagram, comprising:

extracting the shape colimit of the hereditary diagram stored in a memory, the hereditary diagram including a plurality of diagrams;

bringing each of the plurality of diagrams in the hereditary diagram to the shape of the shape colimit to yield a plurality of extended diagrams in the memory; and

taking the colimit of the extended diagrams.

3. The method of claim 2, further comprising: receiving from the user an indication to find the colimit of the hereditary diagram.

4. The method of claim 2, wherein extracting the shape colimit of the hereditary diagram includes:

determining the shape of each of the plurality of diagrams to yield a shape graph in the memory; and

automatically calculating a colimit of the shape diagram.

5. The method of claim 2, further comprising: displaying a representation of the colimit on a display device.

6. The method of claim 5, wherein the representation o the colimit is the name of the colimit.

7. The method of claim 5, wherein the representation of the colimit is a picture of the diagram of the colimit.

8. The method of claim 2, wherein the hereditary diagram includes types of the diagram elements.

9. The method of claim 2, wherein the hereditary diagram includes morphisms between the diagram elements.

10. The method of claim 2, wherein the hereditary diagram is displayed with indicators on its arcs indicating what morphism is associated with the arcs.

11. The method of claim 2, wherein the colimit of the hereditary diagram is displayed with indicators on its arcs indicating that that arcs constitute a cocone colimit.

12. A computer-implemented system of automated software specification, comprising:

specification modules stored as separate entities, with their relations displayed on a computer screen in terms of their specification morphisms, where the specification morphisms translate the specification signatures while preserving the logical structure of the specification;

a portion that determines and displays, in response to a user command, multiple specification diagrams, each of which captures the relation between two or more specification modules, along with its specification morphisms;

a portion that builds and displays, in response to a user command, a diagram of the specification diagrams, the diagram of specification diagrams retaining the diagram morphisms of the specification diagrams; and

a portion that computes the colimits of the hereditary diagram of diagrams to compose large software modules while preserving the decomposition of the involved components.

METHOD AND APPARATUS FOR DETERMINING COLIMITS

OF HEREDITARY DIAGRAMS

ABSTRACT OF THE DISCLOSURE

A computer-implemented method and system for determining colimits of hereditary diagrams. A user specifies a diagram of diagram and specifies performance of a colimit operation. Once the colimit is performed, the name of the colimit is added to the hereditary diagram. The described embodiment supports diagrams of diagrams, also called hierarchical diagrams.

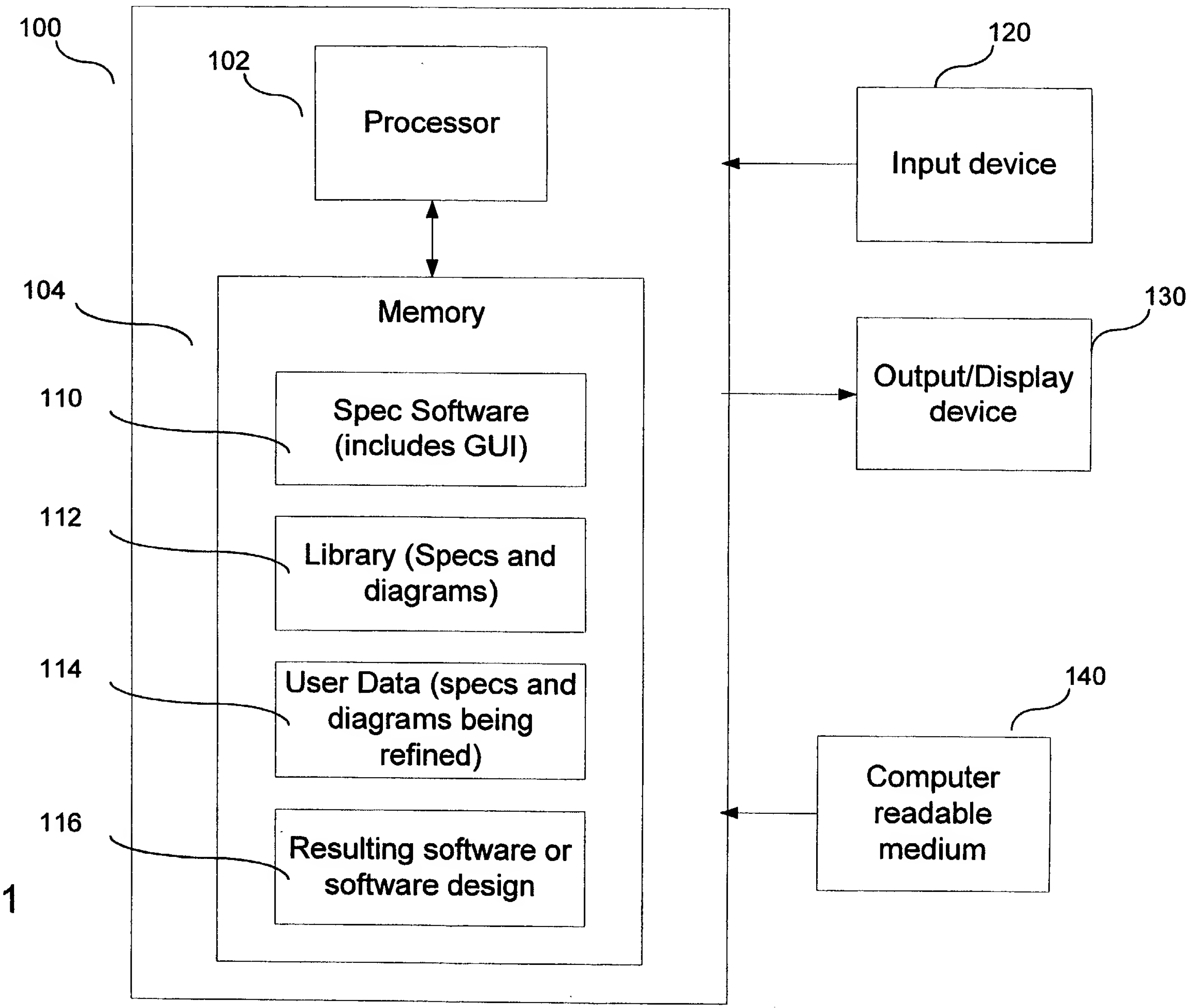

100
102
Processor
104
Memory
110
Spec Software (includes GUI)
112
Library (Specs and diagrams)
114
User Data (specs and diagrams being refined)
116
Resulting software or software design
120
Input device
130
Output/Display device
140
Computer readable medium

Fig. 1

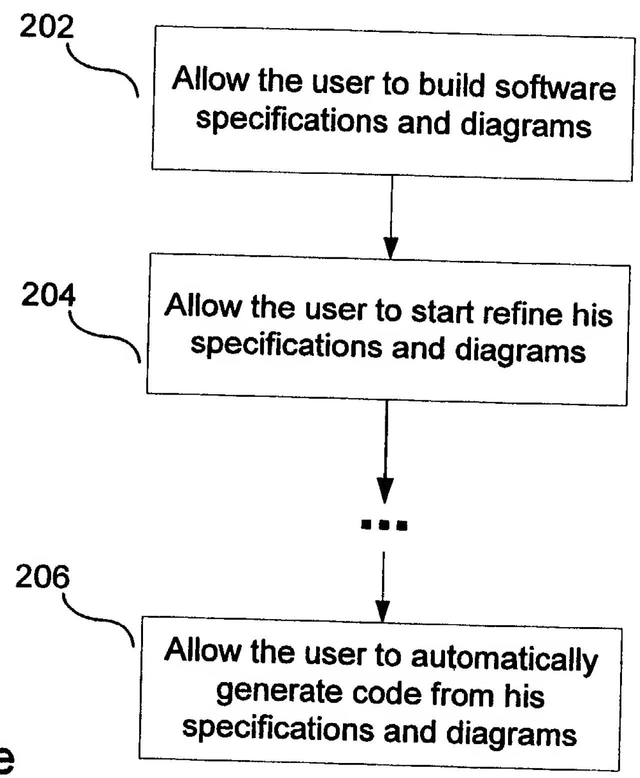


Fig. 2(a)
Spec Software

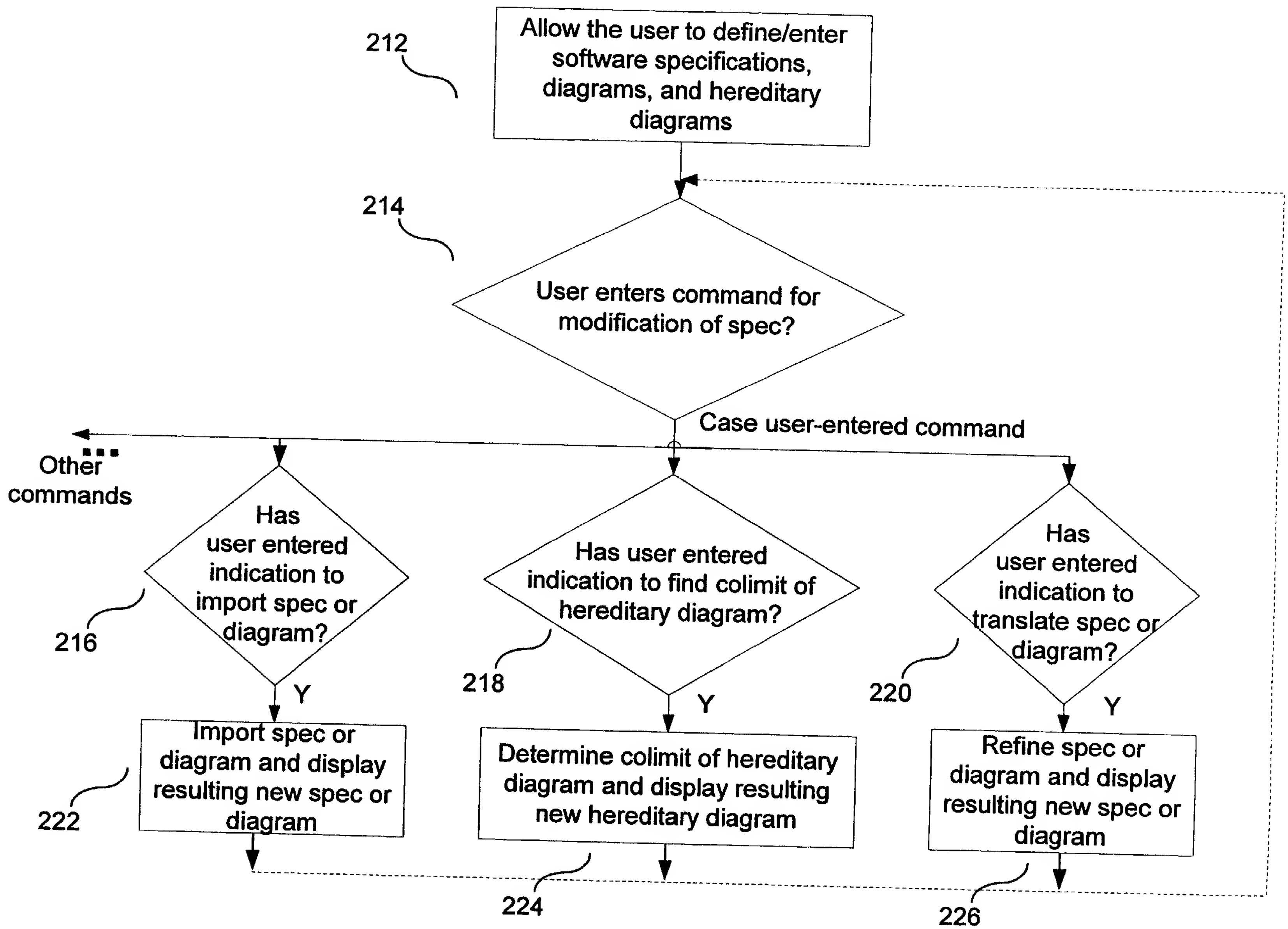
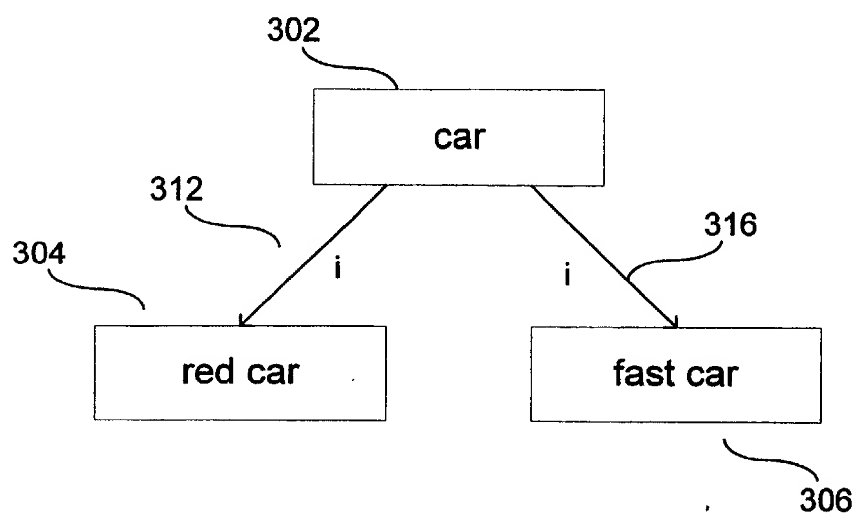
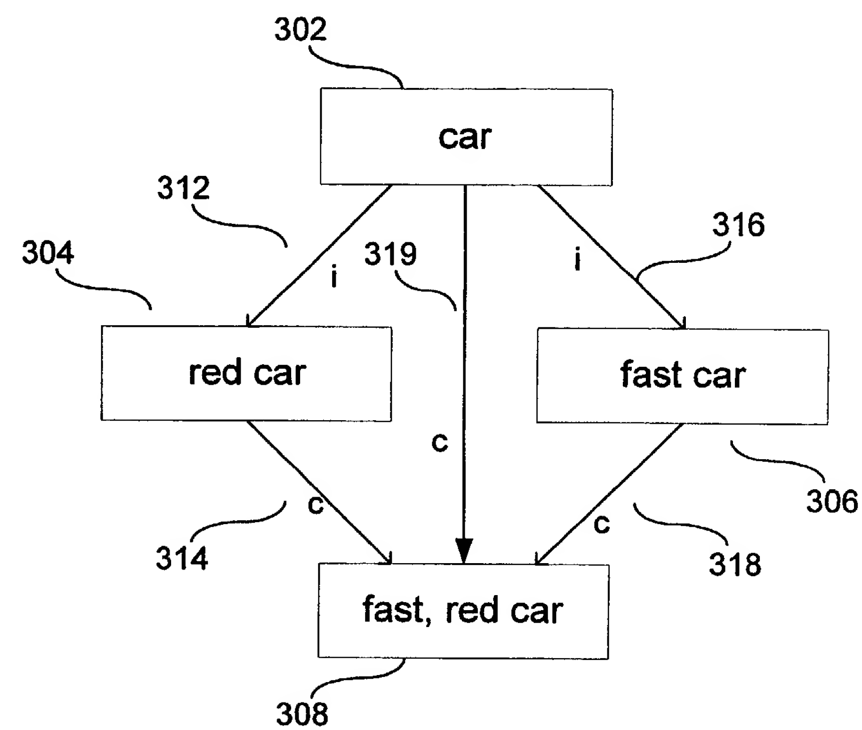


Fig. 2(b)
Spec Software



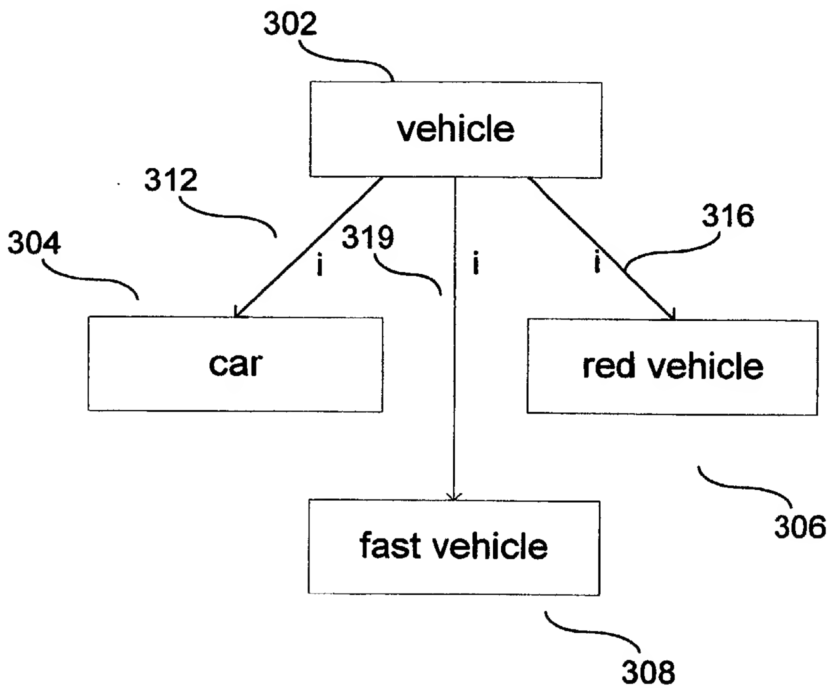
Refining a Specification

Fig. 3(a)

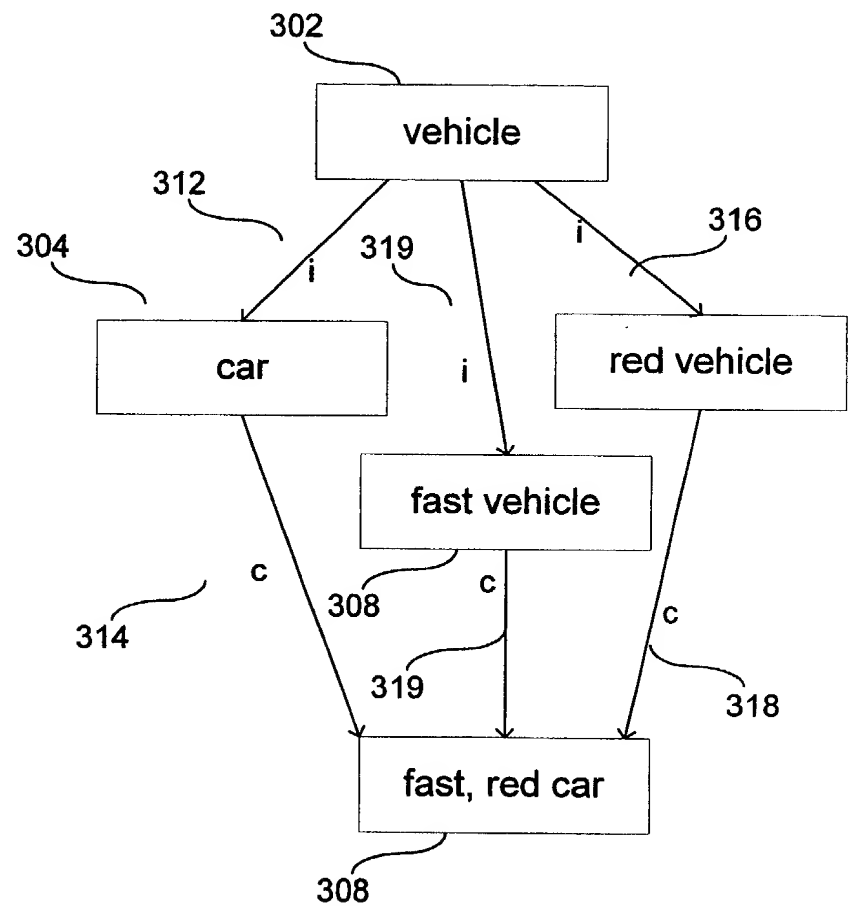


Example of Using a Colimit to Combine Refined Specifications

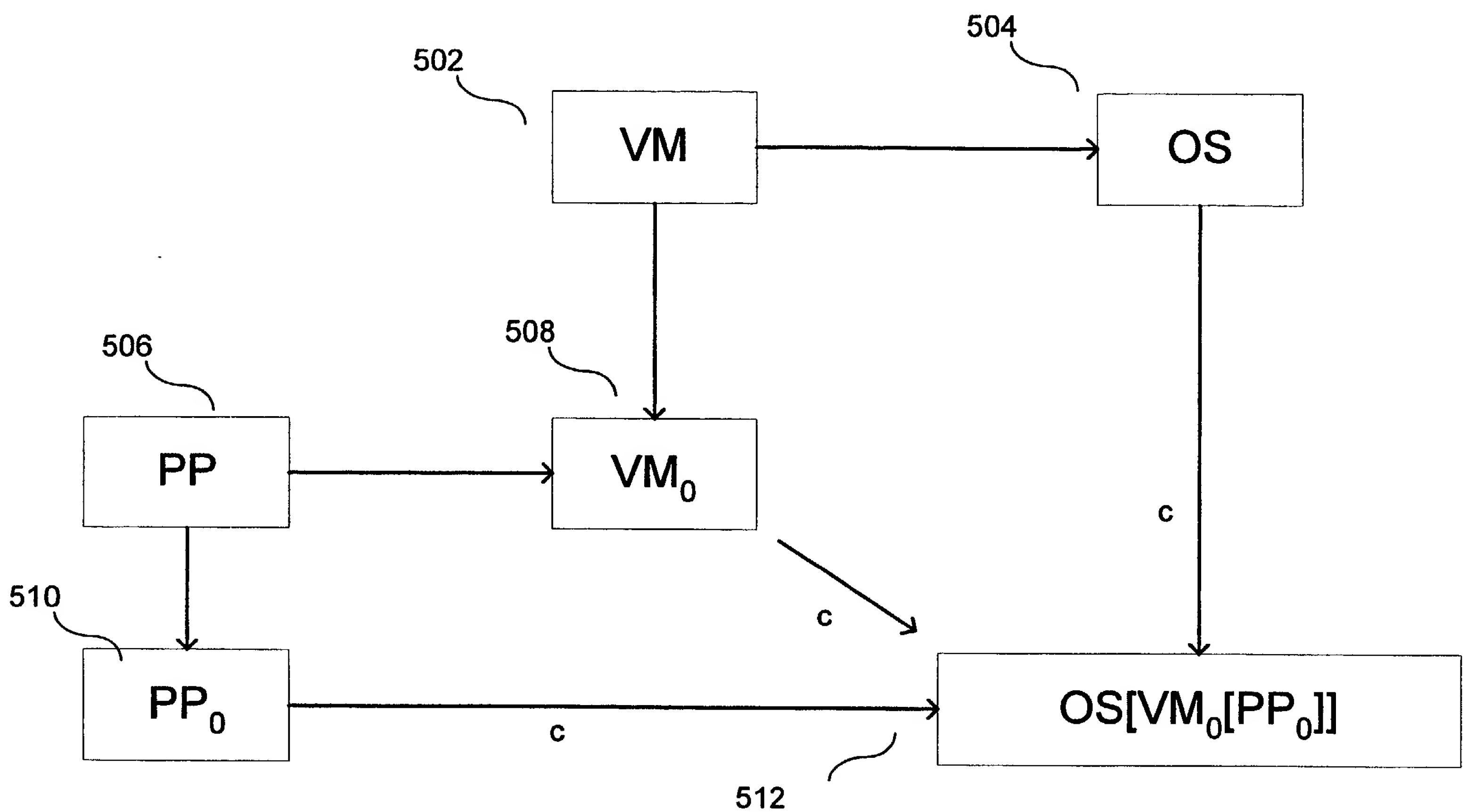
Fig. 3(b)



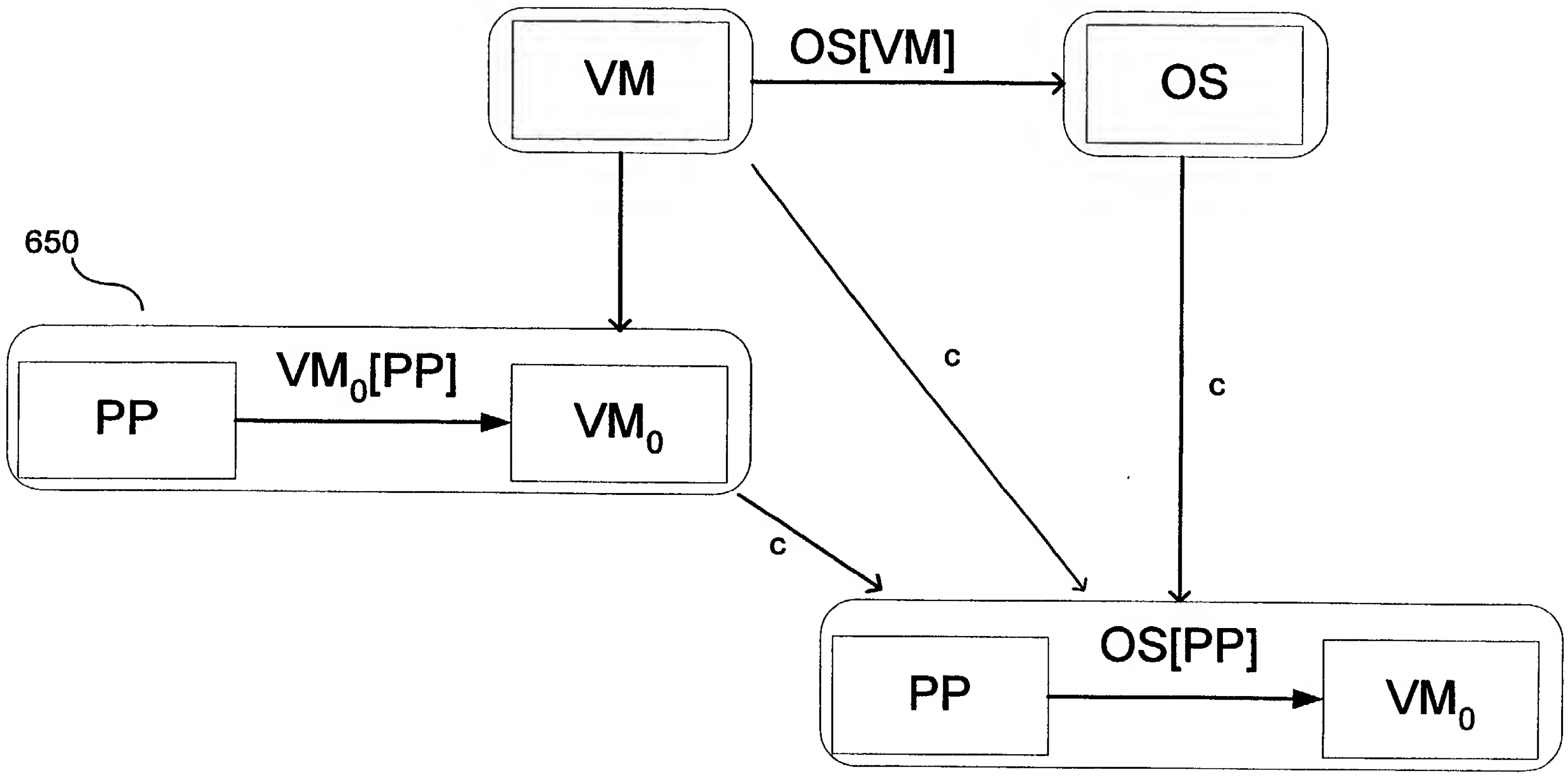
Refining a Specification
Fig. 4(a)



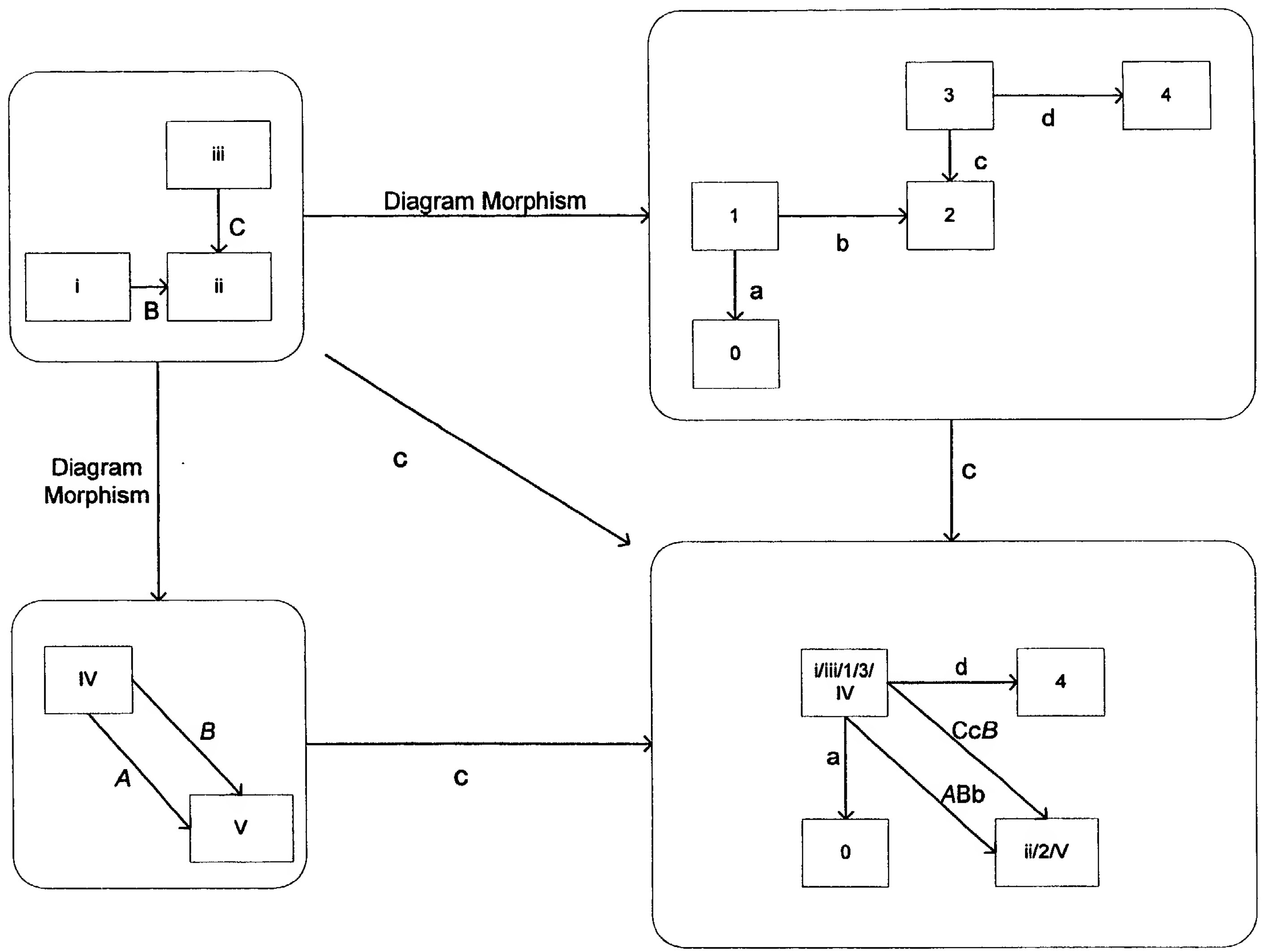
Example of Using a Colimit to
Combine Refined Specifications
Fig. 4(b)



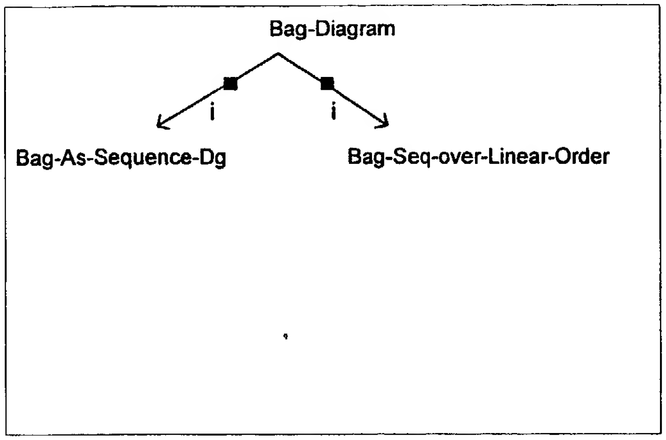
Example Colimit of Specifications
Fig. 5



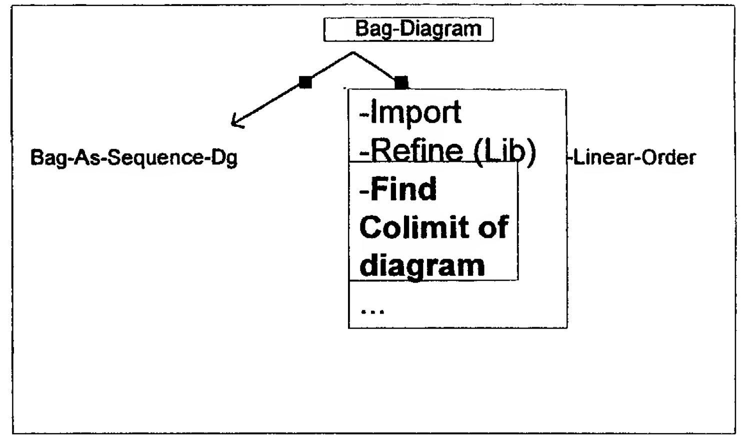
Example Colimit of Diagrams of Diagrams
Fig. 6



Example of Taking the Colimit of Hereditary Diagrams
Fig. 7

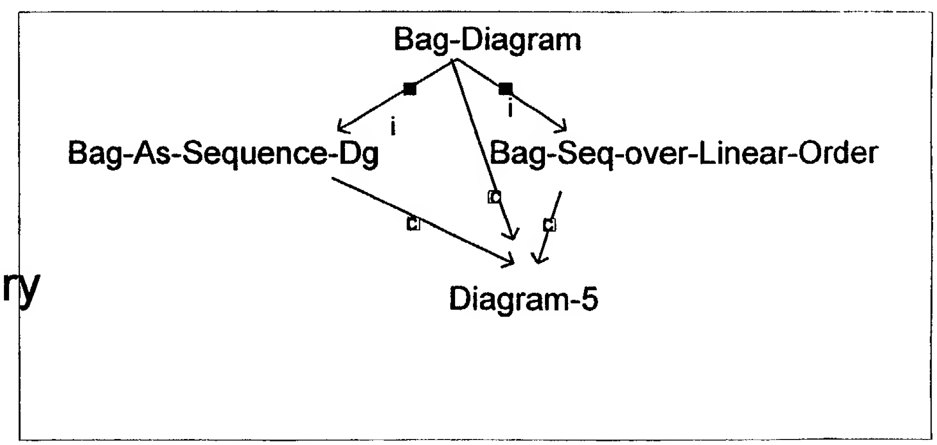


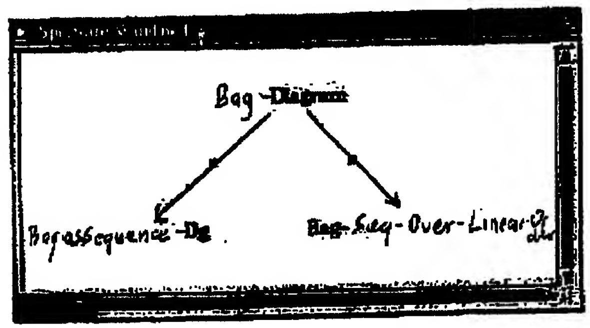
Example user interface showing a hereditary diagram
Fig. 8(a)



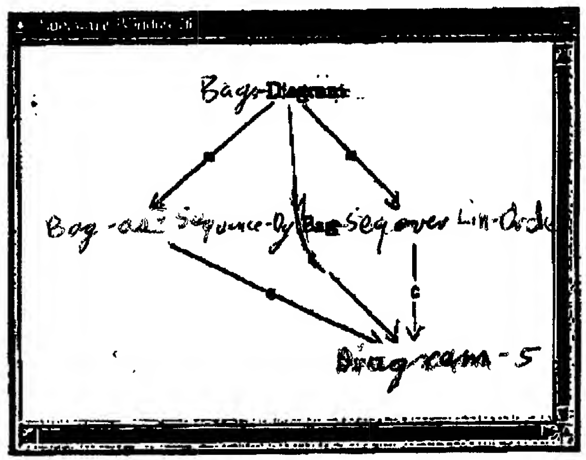
Example user interface showing a hereditary diagram (interface for user to indicate "find colimit" operation)
Fig. 8(b)

Example user interface showing a hereditary diagram after the user indicates a "find colimit" operation for the hereditary diagram and the colimit operation is performed
Fig. 8(c)



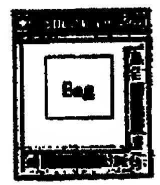


Hereditary diagram
Fig 9(a)



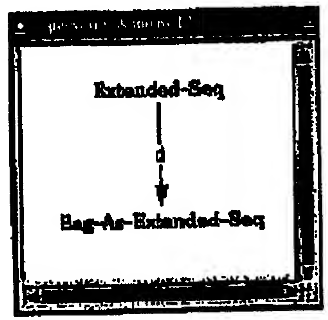
Hereditary diagram, including colimit

Fig 9(b)



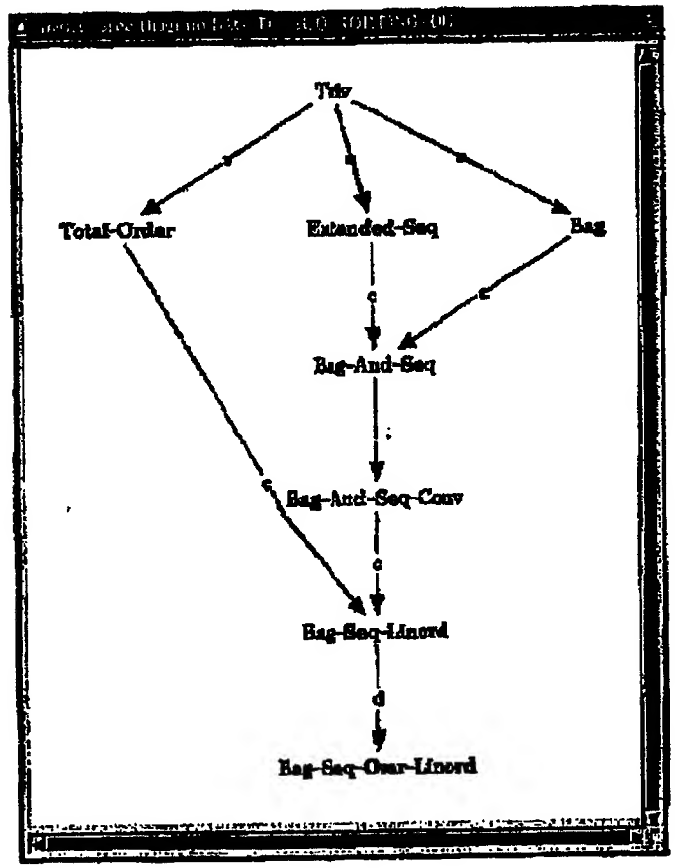
Bag diagram
(obtained by expanding node
Bag-Diagram
in Hereditary diagram)

Fig 9(c)



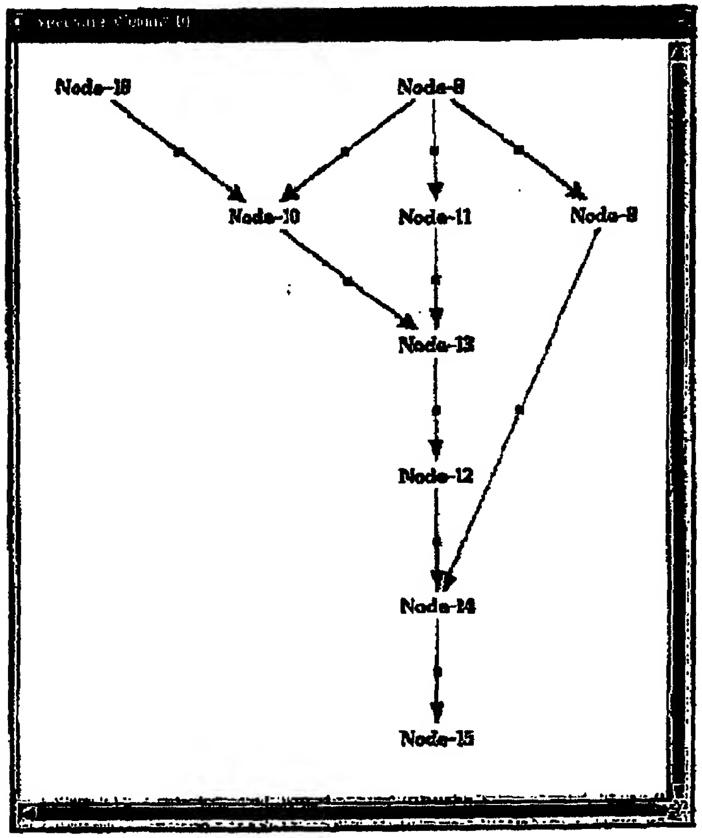
Bag-as-Sequence diagram
(obtained by expanding node
Bag-as-Sequence-diagram
in Hereditary diagram)

Fig 9(d)

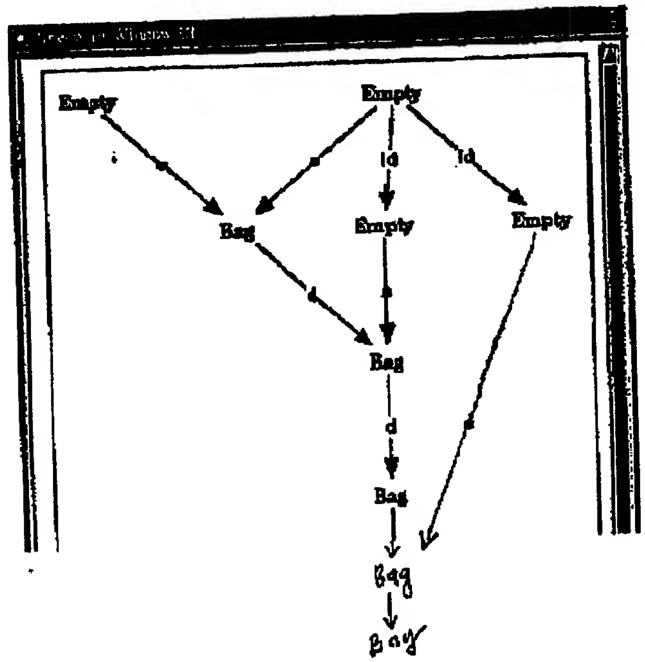


Bag-Seq-over-Linear-Order diagram
(obtained by expanding node
Bag-Seq-over-Linear-Order-diagram
in Hereditary diagram)

Fig 9(e)

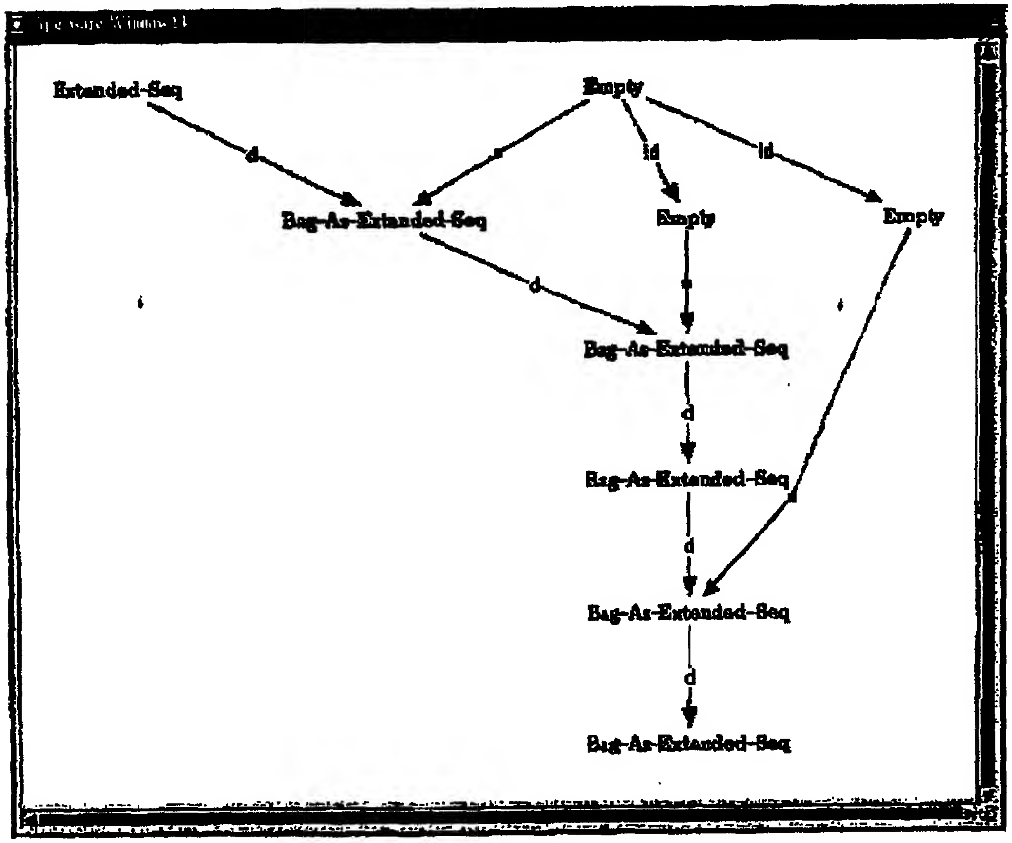


Shape of colimit
Fig 9(f)



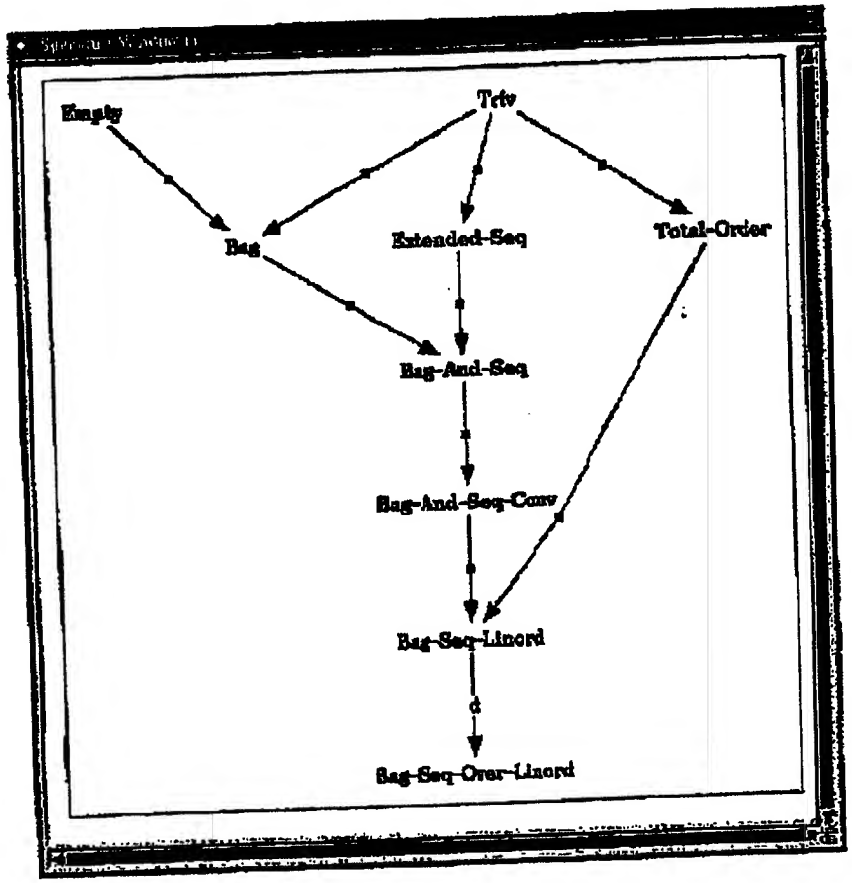
Extended Bag diagram

Fig. 9(g)



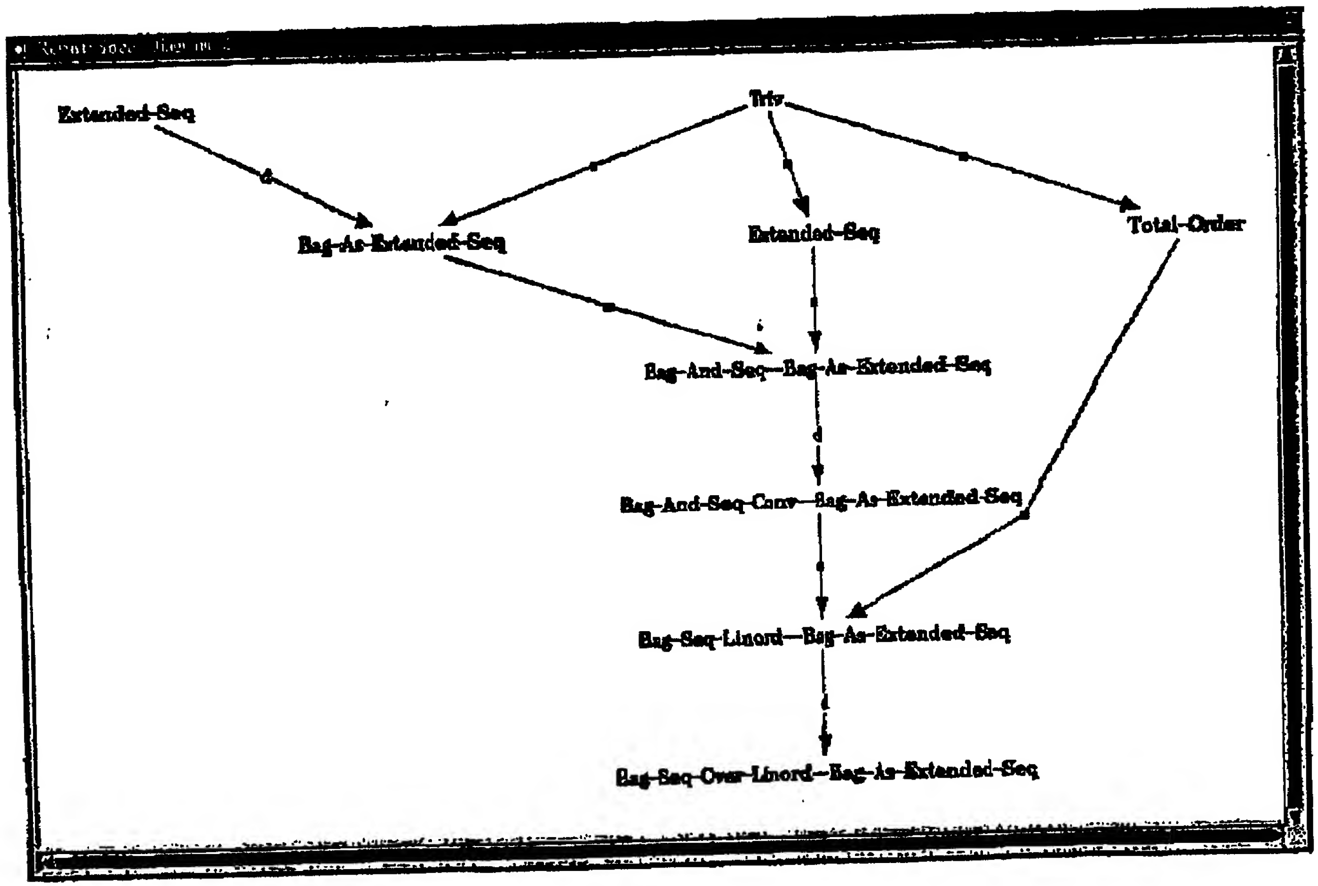
Extended Bag-as-Sequence diagram

Fig 9(h)

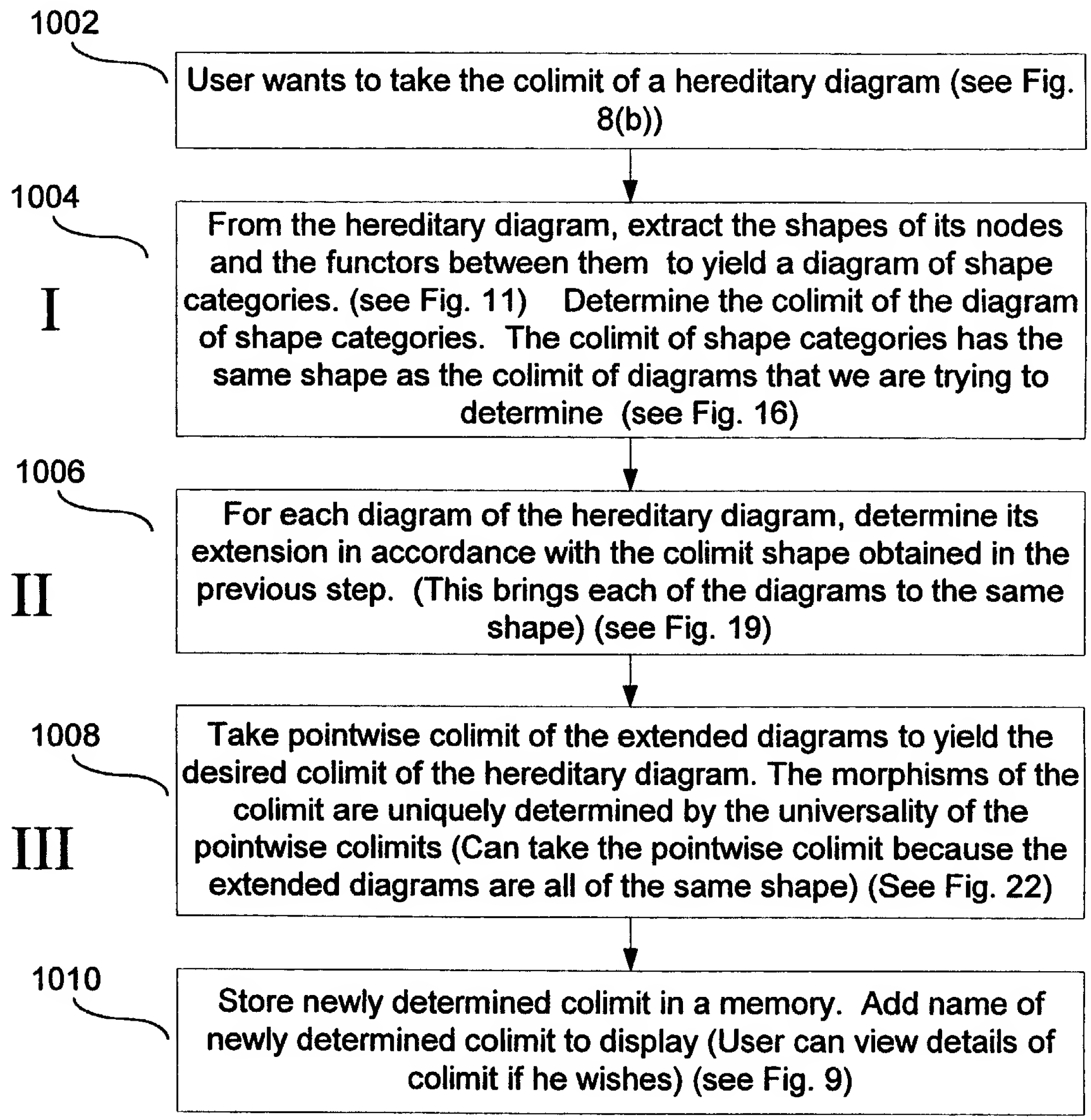


Extended Bag-Seq-over-Linear-Order diagram

Fig 9(i)

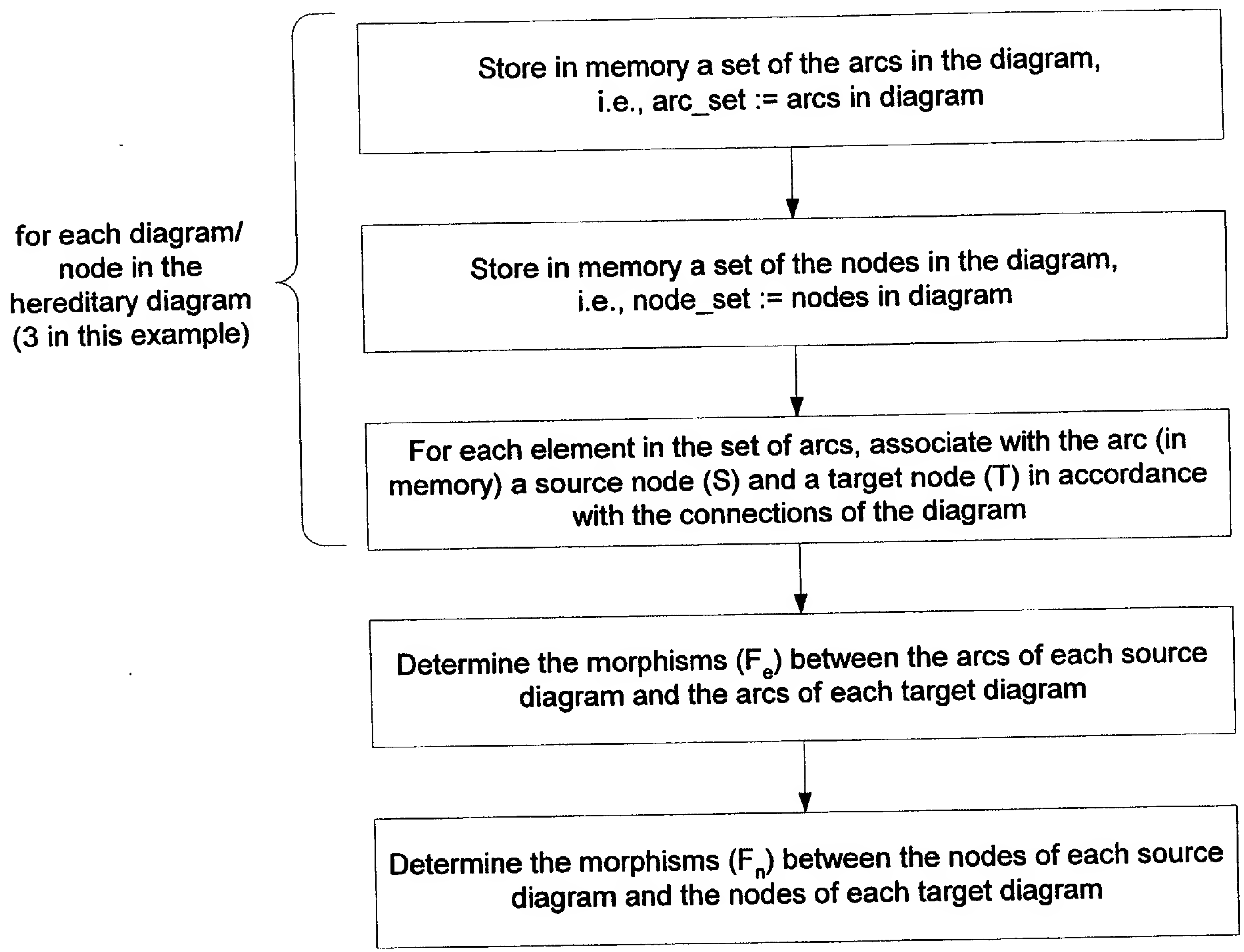


Colimit of Hereditary diagrams
(final result)
Fig 9(j)



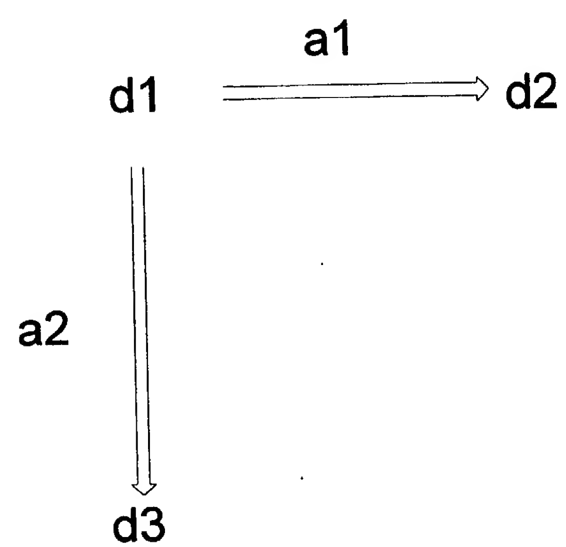
Finding a Colimit of Hereditary Diagrams

Fig. 10



PART I: Extract the shapes and shape functors to yield a diagram of shape categories

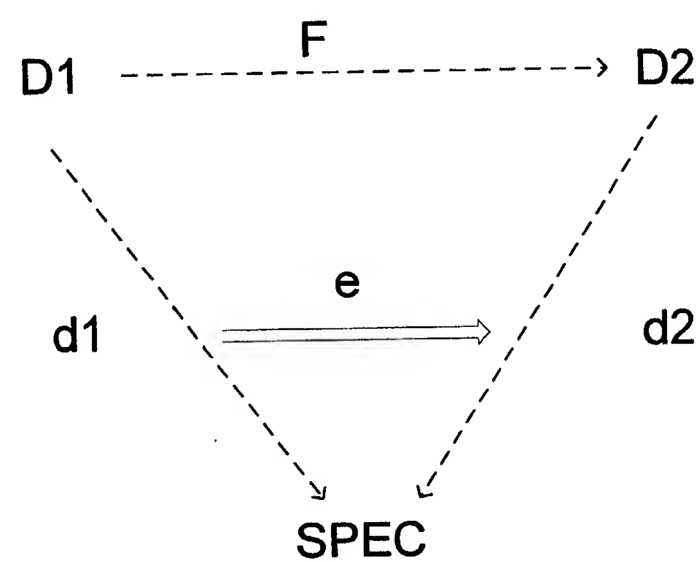
Fig. 11



A Hereditary Diagram: Three Diagrams and Two Arcs.

Each arc a1 and a2 represents a shape morphism having 1) a shape functor (such as F) and 2) a natural shape transformation (such as e: d1 --> d2)

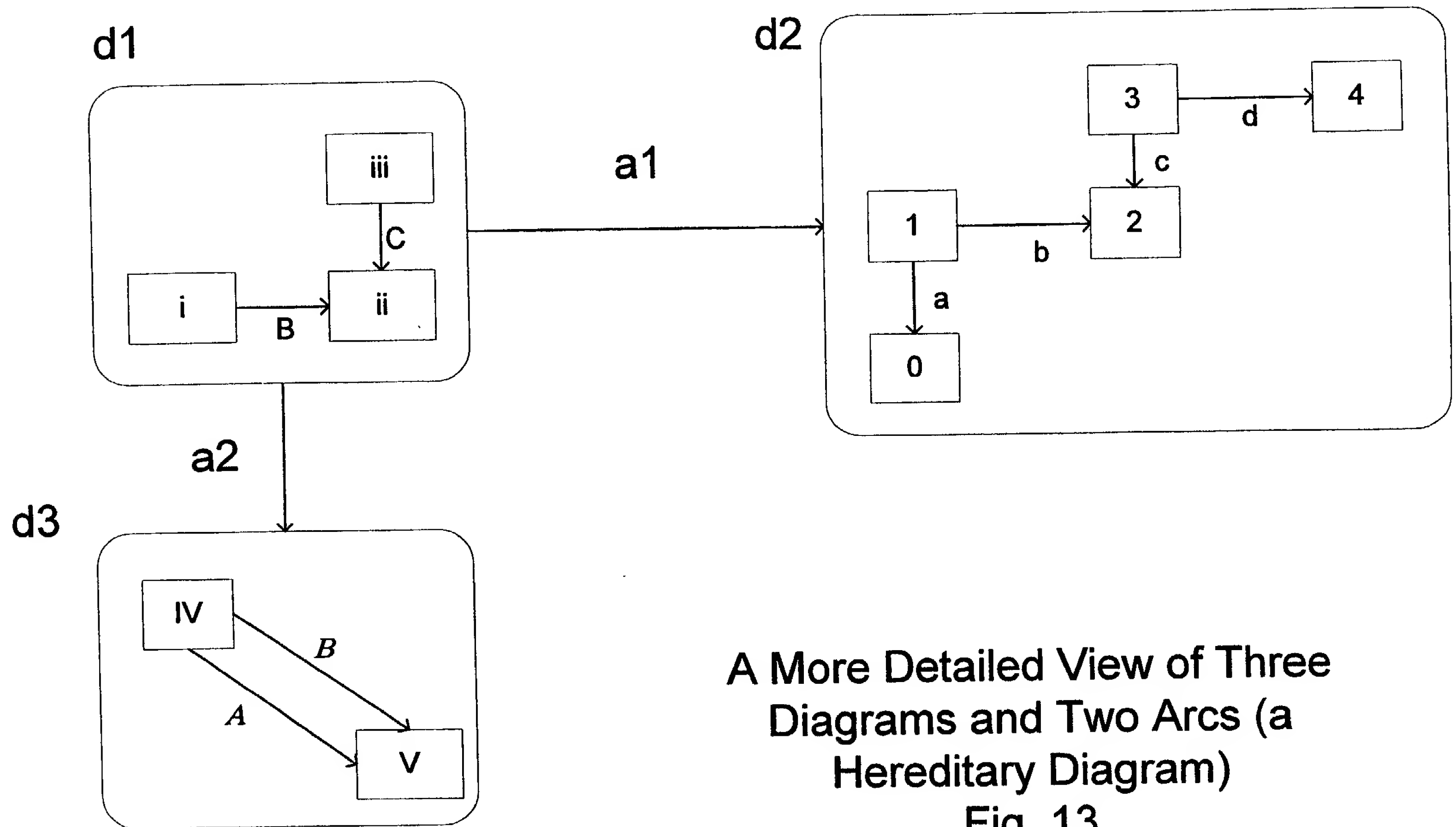
Fig. 12(a)



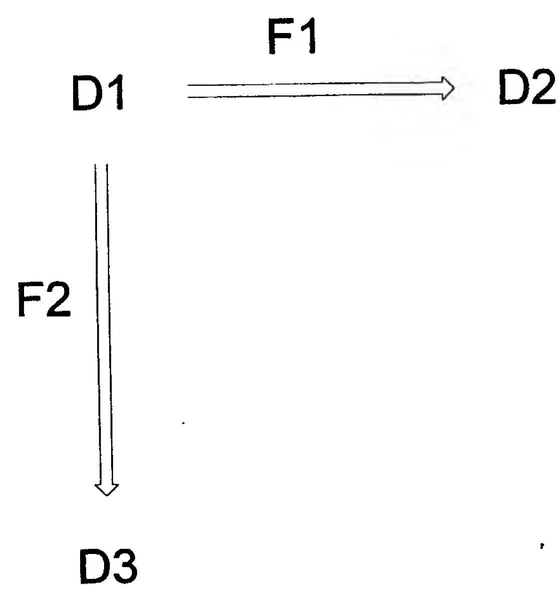
A Shape Morphism

where d1 and d2 are diagrams,
F is a shape functor,
e is a natural transformation from d1 to (d2 composed with F)
D1 and D2 are shape categories of diagrams, and SPEC is the category Spec

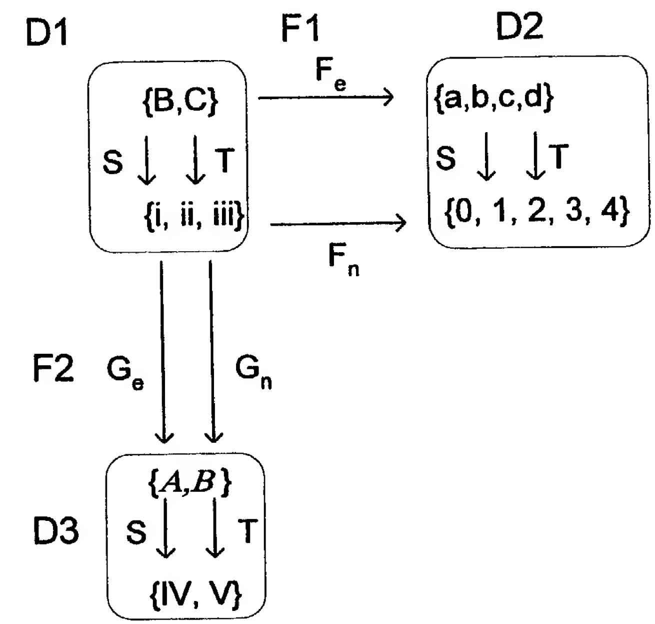
Fig. 12(b)



A More Detailed View of Three Diagrams and Two Arcs (a Hereditary Diagram)
Fig. 13



Extract the
Shapes and
Shape Functors
(D1 is shape of
diagram d1, F1 is
shape functor)
Fig. 14



More Detailed View of Extracting the
Shapes and Shape Functors
(continued on Figs. 15(b)-15(d))
Fig. 15(a)

Arcs: B -> b
C -> c
Nodes: i -> 1
ii -> 2
iii -> 3

Mapping for F1
Fig. 15(b)

Arcs: B -> *A*
C -> *B*
Nodes: i -> IV
ii -> V
iii -> IV

Mapping for F2
Fig. 15(c)

Arc	B	C	a	b	c	d	*A*	*B*
Source	i	iii	1	1	3	3	IV	IV
Target	ii	ii	0	2	2	4	V	V

Source (S) and Target (T) Functions for Hereditary Diagrams

Fig. 15(d)

Compute colimit of sets of nodes: Store in memory a disjoint union of all nodes (ignore arcs). Determine the equivalence relations identifying those nodes that are connected by some arc of the hereditary diagram. All nodes of the diagrams that fall in the same equivalence class are identified as a single node in the colimit

↓

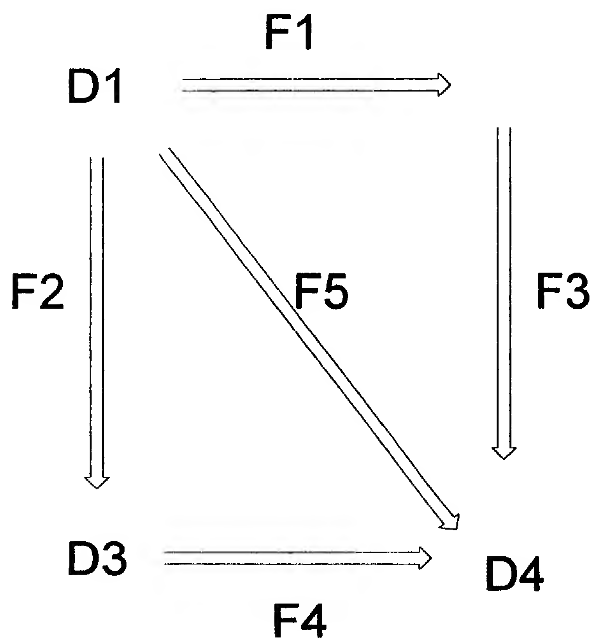
Compute colimit of sets of arcs: Store in memory a disjoint union of all arcs (ignore nodes). Determine the equivalence relations identifying those arcs that are connected by some arc of the hereditary diagram. All arcs of the diagrams that fall in the same equivalence class are identified as a single arc in the colimit

↓

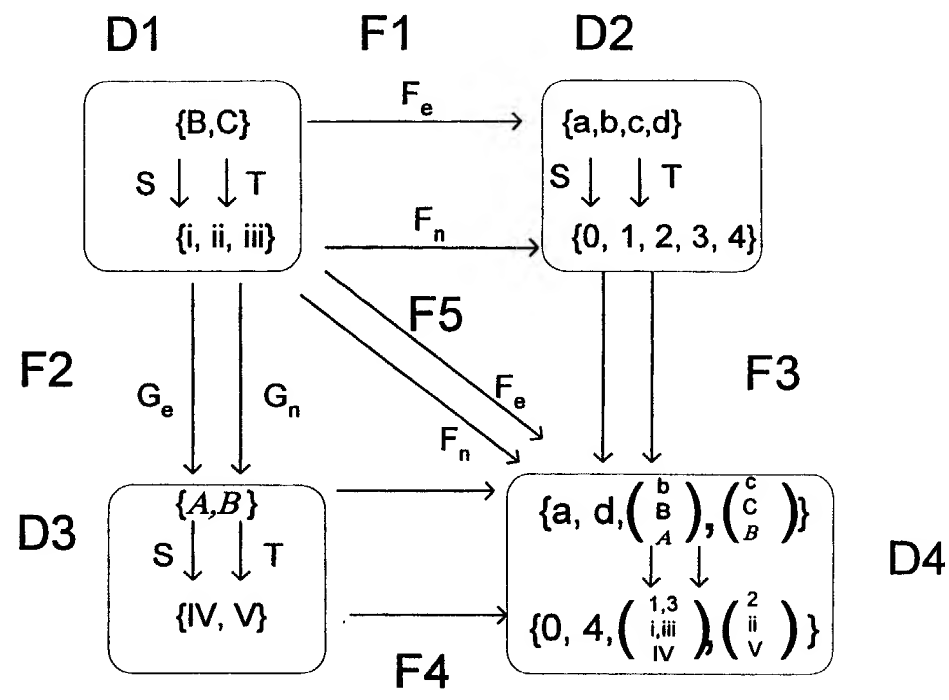
Consider the relationship between the equivalence classes of arcs and nodes. For each arc in the colimit, the universal property of the colimit (of the sets of arcs) determines a source node (S) and a target node (T)

PART I: Determine the colimit of the diagram of shape categories.

Fig. 16



More Detailed View of Taking the Colimit
Fig. 17

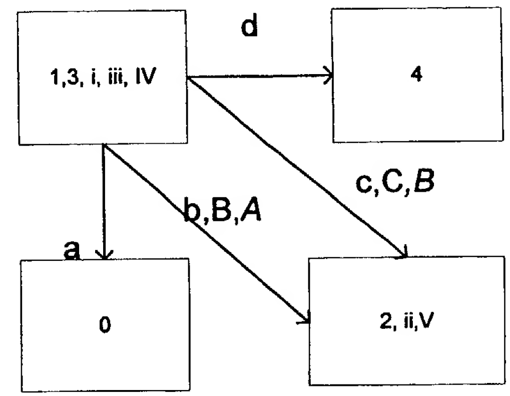


More Detailed View of Taking the Colimit
(See Figs 18(b)-(f))

Fig. 18(a)

Arc	a	d	b A *A*	c C *B*
Source	1,3 i,iii IV	1,3 i,iii IV	1,3 i,iii IV	1,3 i,iii IV
Target	0	4	2 ii V	2 ii V

Source (S) and Target (T) Functions for Shape Colimit D4
Fig. 18(b)



The Colimit D4 of the Shape Diagrams
Fig. 18(c)

Arcs: a -> a
d -> d
b -> b,B,*A*
c -> c,C,*B*

Nodes: 0 -> 0
1 -> 1,3, i,iii, IV
2 -> 2, ii,V
3 -> 1,3, i,iii, IV
4 -> 4

Mapping for F3
Fig. 18(d)

Arcs: *A* -> b,B,*A*
B -> c,C,*B*

Nodes: IV -> 1,3, i, iii, IV
V -> 2, ii,V

Mapping for F4
Fig. 18(e)

Arcs: B -> b,B,*A*
C -> c,C,*B*

Nodes: i -> 1,3, i, iii, IV
ii -> 2, ii,V
iii -> 1,3, i, iii, IV

Mapping for F5
Fig. 18(f)

For each node n in colimit D4:

↓

Find nodes s in shape diagram (for example diagram D1) that have a path i to the node n. Yields a set of pairs : {<s,i> | i is a path from F(s) to node n in D4}

↓

For two pairs in the set (for example: $<s_i,i>$ and $<s_j,j>$)

↓

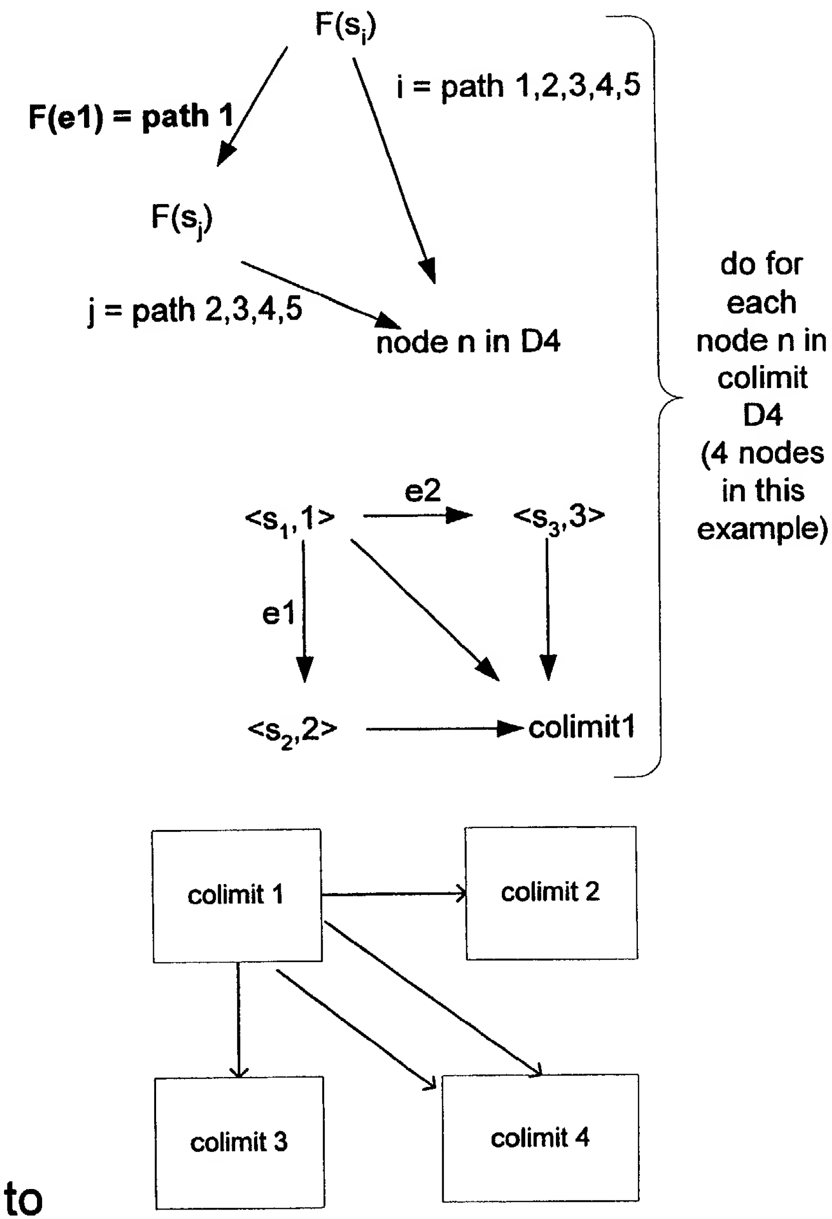
By definition, s_i and s_j both connect to the same node in colimit D4. Find a path e between s_i and s_j

↓

Once each path F(e) has been found between each pair in the set, make a graph and take the colimit. This colimit is the image of node n in the extended diagram

↓

Each arc in D4 is uniquely defined and determined using the universality of the colimits for the nodes in the extended diagram



extended diagram from example

PART II:Extending one Diagram (repeat to extend each diagram)

Fig. 19

Extension for Diagram d1:

$$\begin{tikzcd}[column sep=small]
D1 \arrow[rr] \arrow[dr, Rightarrow, "F5"'] & & SPEC \\
& D4 \arrow[ur, Rightarrow, "Lf5d1"'] &
\end{tikzcd}$$

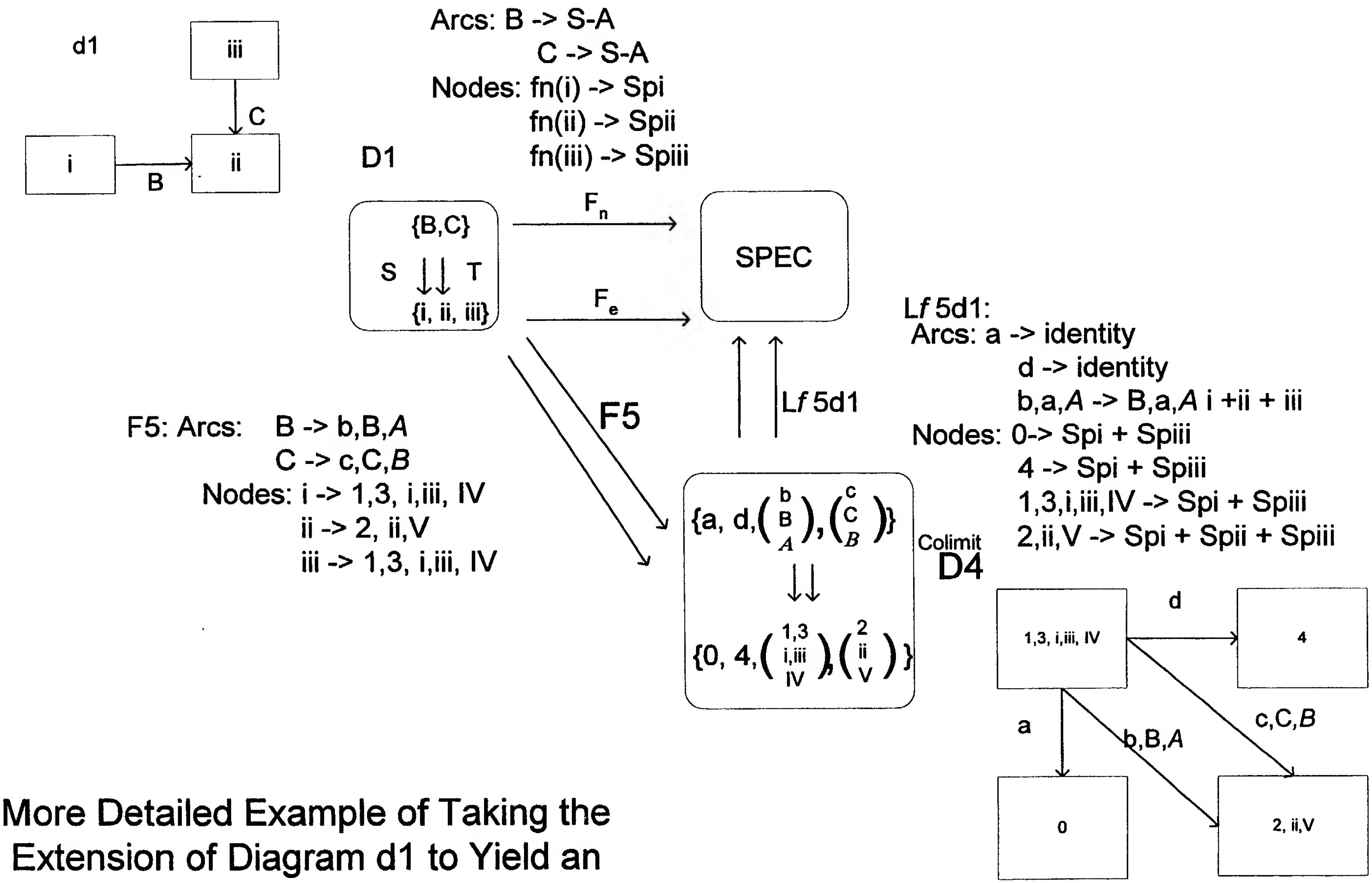
Extension for Diagram d2:

$$\begin{tikzcd}[column sep=small]
D2 \arrow[rr] \arrow[dr, Rightarrow, "F3"'] & & SPEC \\
& D4 \arrow[ur, Rightarrow, "Lf3d2"'] &
\end{tikzcd}$$

Extension for Diagram d3:

$$\begin{tikzcd}[column sep=small]
D3 \arrow[rr] \arrow[dr, Rightarrow, "F4"'] & & SPEC \\
& D4 \arrow[ur, Rightarrow, "Lf4d3"'] &
\end{tikzcd}$$

Example of Taking the
Extension of Each Node of the
Hereditary Diagram
Fig 20



More Detailed Example of Taking the Extension of Diagram d1 to Yield an Extended Diagram

Fig. 21(a)

$i \xrightarrow{F_5} \begin{pmatrix} 1,3 \\ i, iii \\ \mathrm{IV} \end{pmatrix}$

$\downarrow a$

Node 0

(1)

For four Nodes in D4

$\{ \langle i, a \rangle \mid a : F_5(i) = \begin{pmatrix} 1,3 \\ i\ iii \\ \mathrm{IV} \end{pmatrix} \xrightarrow{a} 0 \}$

$\langle iii, a \rangle \mid a : F(iii) = \begin{pmatrix} 1,3 \\ i, iii \\ \mathrm{IV} \end{pmatrix} \xrightarrow{a} 0 \}$

$\langle i, a \rangle \rightarrow 0 \leftarrow \langle iii, a \rangle$, id

$\langle i, a \rangle \longrightarrow Sp_i + \langle iii, a \rangle \longrightarrow Sp_{iii}$

coproduct cuz no arc between them

Node 4

$\{ \langle i, d \rangle \mid d : F_5(i) = \begin{pmatrix} 1,3 \\ i, iii \\ \mathrm{IV} \end{pmatrix} \xrightarrow{d} 4 \}$

$\langle iii, d \rangle \mid$ —— —— $\}$

same as above

Node $\begin{pmatrix} 1,3 \\ i, iii \\ \mathrm{IV} \end{pmatrix}$

$\{ \langle i, id \rangle$

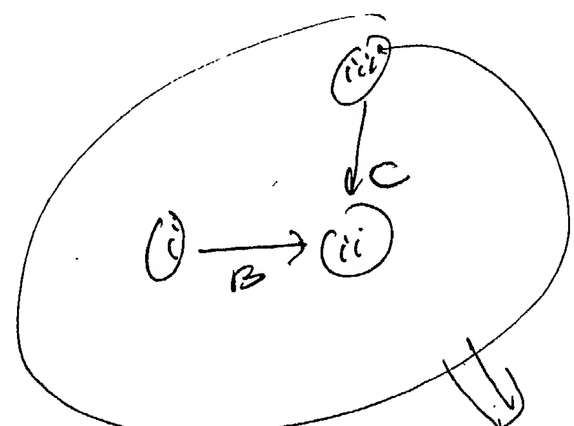
$\langle iii, id \rangle \}$

Node $\begin{pmatrix} 2 \\ ii \\ \mathrm{V} \end{pmatrix}$

$\{ (i, BA)$

$(iii, CB) \}$

(ii, id)



$Sp_{i+ii+iii}$

colimit

Fig 21(b) Example of Diagram Extension

Fig 15 ②

for arcs in D4

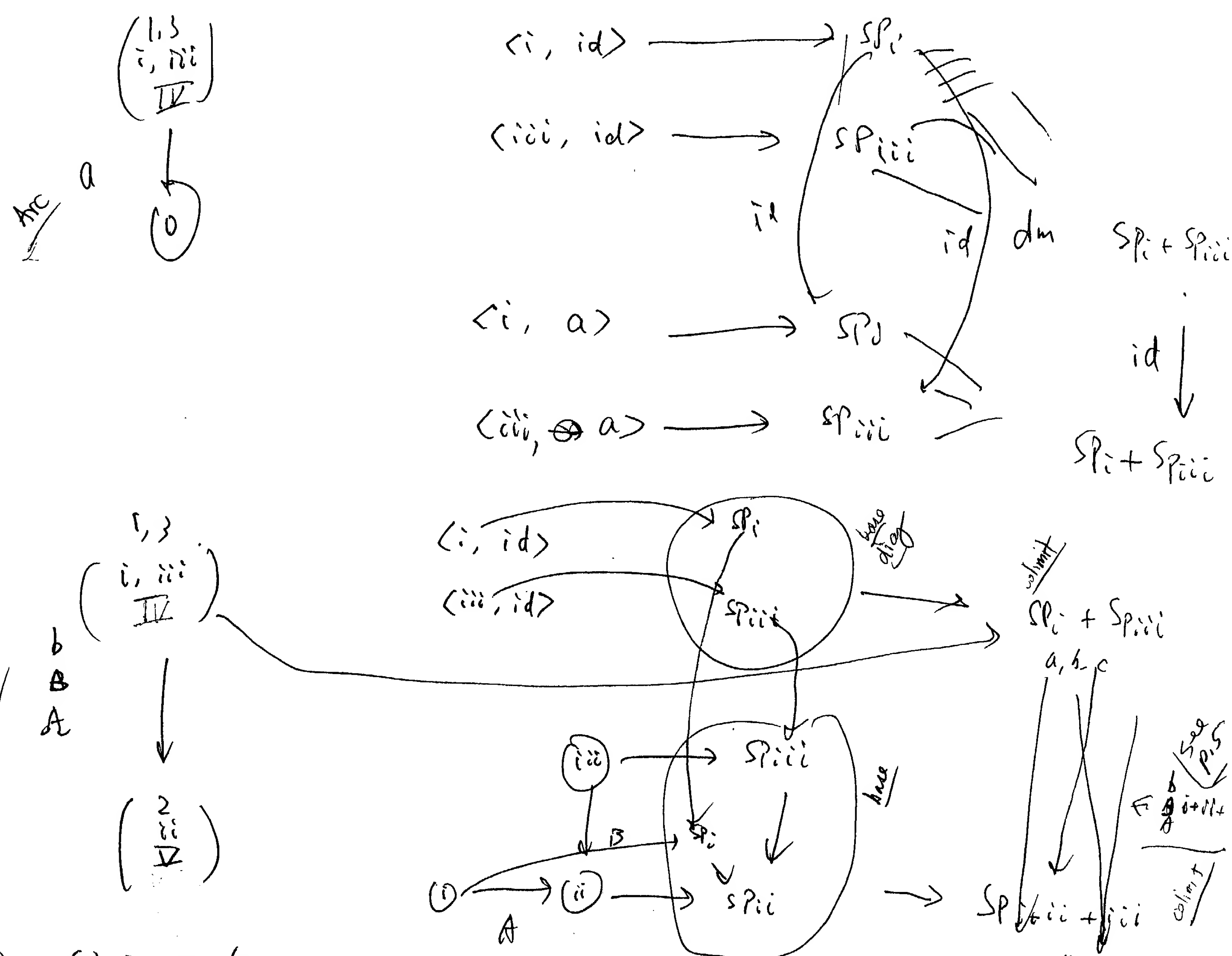


Fig 21(c) Example of Diagram extension(cont)

$$Sp_i = \left\{ \begin{array}{l} \text{spec } Sp_i \text{ is} \\ \quad \text{sort } S_1 \\ \quad \text{op } f_1 : S_1 \to \text{boolean} \\ \quad \text{op } f_2 : S_1 \to \text{boolean} \\ \quad \text{axiom } f_1 \to f_2 \text{ is} \\ \qquad f_1(x) \Rightarrow f_2(x) \end{array} \right\}$$

$$\begin{tikzcd}
& iii \arrow[d, "B"] \arrow[rr] & & Sp_{iii} \arrow[d, "S\text{-}B"] \\
i \arrow[r, "A"] \arrow[drr, bend right] & ii \arrow[rr] \arrow[dr] & & Sp_{ii} \\
& & Sp_i \arrow[ur, "S\text{-}A"'] &
\end{tikzcd}$$

Base diagram

$$Sp_{ii} = \left\{ \begin{array}{l} \text{spec } Sp_{ii} \text{ is} \\ \quad \text{sort } S_2 \\ \quad \text{op } g : S_2 \to \text{boolea} \end{array} \right\}$$

$$Sp_{iii} = \left\{ \begin{array}{l} \text{spec } Sp_{iii} \text{ is} \\ \quad \text{sort } S_1 \\ \quad \text{op } f_1 : S_1 \to \text{boolen} \end{array} \right\}$$

$$S\text{-}A = \left\{ \begin{array}{lcl} Sp_i & \longrightarrow & Sp_{ii} \\ S_1 & \longrightarrow & S_2 \\ f_1 & \longrightarrow & g \\ f_2 & \longrightarrow & g \end{array} \right\}$$

$$S\text{-}B = \left\{ \begin{array}{lcl} Sp_{iii} & \longrightarrow & Sp_{ii} \\ S_1 & \longrightarrow & S_2 \\ f_1 & \longrightarrow & g \end{array} \right\}$$

Fig 21(d) Example of diagram [illegible] (cont)

3)

(c)

$$
\begin{tikzcd}
 & Sp_{iii} \arrow[d, "S-B"] \arrow[ddr, bend left, "C_{iii}"] & \\
Sp_i \arrow[r, "S-A"] \arrow[drr, "C_i"'] & Sp_{ii} \arrow[dr, "C_{ii}"] & \\
 & & Sp_{i+ii+iii}
\end{tikzcd}
$$

$$C_i = \left\{ \begin{array}{l} Sp_i \rightarrow Sp_{i+ii+iii} \\ S_1 \longrightarrow S_2 \\ f_1 \longrightarrow g \\ f_2 \longrightarrow g \end{array} \right\}$$

$$C_{ii} = \left\{ \quad id \quad \right\}$$

$$C_{iii} = S-B$$

~~colimit~~ $Sp_{i+ii+iii} =$ {

Spec $Sp_{i+ii+iii}$ is

sort ~~Sort~~ Sp_i - S_1 S_2

Sort Sp_{iii}, S_1

Sort S_2

op g : $S_2 \rightarrow$ boolean

axiom g-g is

$g(x) \Rightarrow g(x)$

}

Fig 21(e) Example of Diagram Extension (cont)

$$Sp_i + Sp_{iii} = \left\{ \begin{array}{l} \text{sort } Sp_i.S_1 \\ \text{sort } Sp_{iii}.S_1 \\ \text{op } Sp_i.f_1 : Sp_i.S_1 \to \text{boolean} \\ \text{op } Sp_{iii}.f_1 : Sp_{iii}.S_1 \to \text{boolean} \\ \text{op } Sp_i.f_2 : Sp_i.S_1 \to \text{boolean} \end{array} \right.$$

$$\Big\downarrow \; \underset{A}{\overset{b}{\cancel{B}}}{}_{i+ii+iii}$$

$$Sp_{i+ii+iii} = \left\{ \text{See [illegible]} \right\}$$

$$\underset{A}{\overset{b}{\cancel{B}}}{}_{i+ii+iii} = \left\{ \begin{array}{l} Sp_i.S_1 \longrightarrow S_2 \\ Sp_{iii}.S_1 \longrightarrow S_2 \\ Sp_i.f_1 \longrightarrow f \\ Sp_{iii}.f_1 \longrightarrow g \\ Sp_i.f_2 \longrightarrow g \end{array} \right\}$$

Fig 21(F) Example of Diagram Extension (cont)

After finishing the extension for each diagrams, let us use the following example:

Original diagrams:

$$\begin{tikzcd}
\boxed{d1:\ D1 ===> SPEC} \arrow[r, "d12"] \arrow[d, "d13"] & \boxed{d2:\ D2 ===> SPEC} \\
\boxed{d3:\ D3 ===> SPEC} &
\end{tikzcd}$$

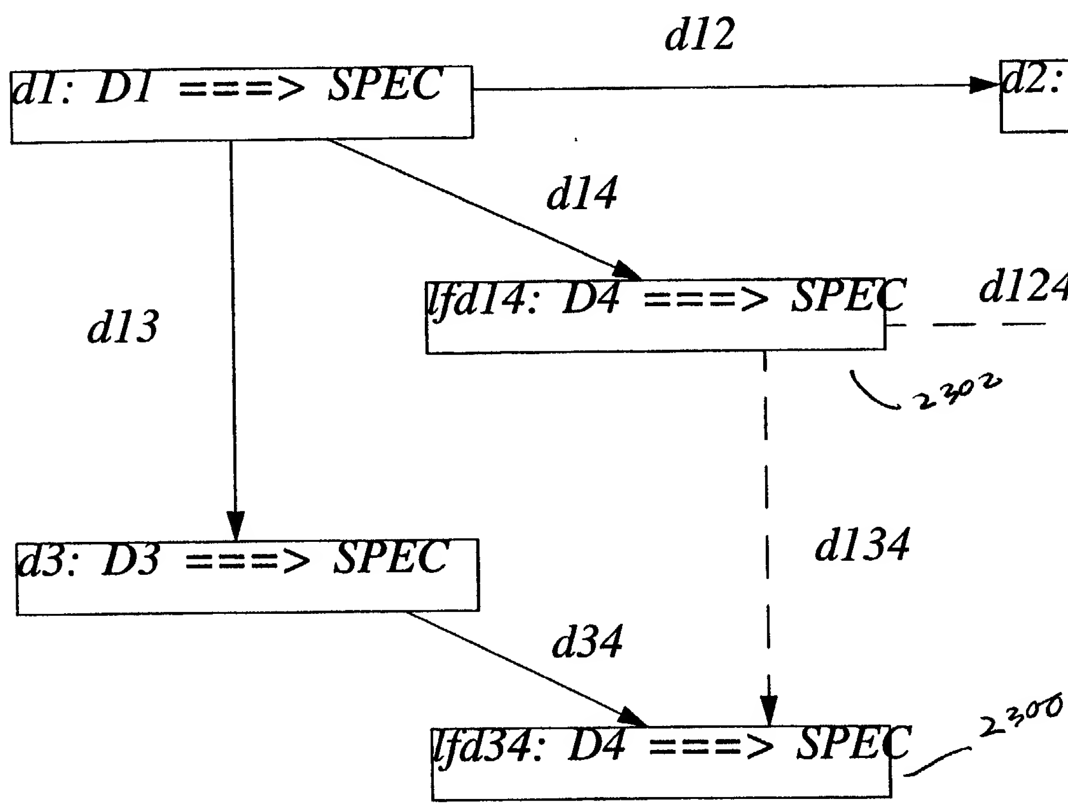
Its underlying shape categories, shape functors and the colimit are:

$$\begin{tikzcd}
\boxed{D1} \arrow[r, "F12"] \arrow[d, "F13"'] \arrow[dr, "F14"] & \boxed{D2} \arrow[d, "F24"] \\
\boxed{D3} \arrow[r, "F34"] & \boxed{D4}
\end{tikzcd}$$

Part III

Fig. 22

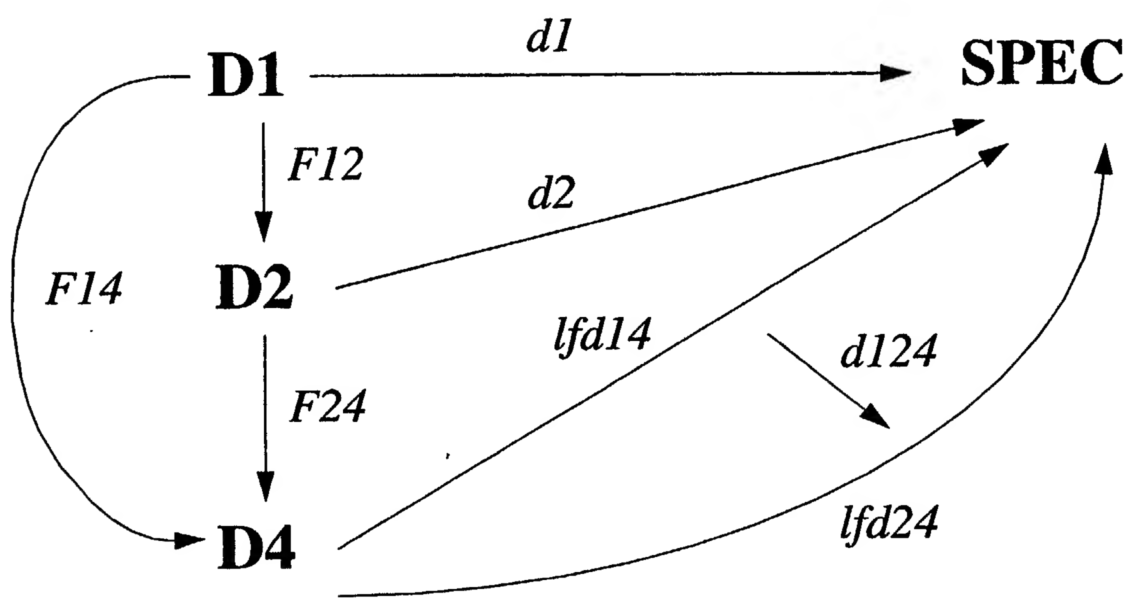
Extended diagrams:



The last algorithm step we are missing for constructing the diagram colimits is the diagram morphisms between extended diagrams. For example, the diagram morphism d124 and d134 (dotted lined arrows in above figure) are the ones needed.

Suppose lfd14 and lfd24 are two extensions of d1 and d2, given the colimit of the shape categories as D4. We would have the following picture

Fig, 23



A morphism between lfd14 and lfd24 is a natural transformation which maps each node of D4 to an arrow in SPEC. We do th y the universal construction of witness arrows.

For any node ni in D4, we have F14(ni) = F12 o F24(ni). Let Sp1ni and Sp2ni two shape categories used for constructing a mapping for ni in its extension of d1 and d2, respectively, then we can have a shape function between Sp1ni and Sp2ni (inclusion, basically). That induces a diagram morphism between the base diagrams for the target of ni in lfd14 and lfd24, respectively. By imposing that diagram morphism and cocone morphism, we can get an unique arrow between lfd14(ni) and lfd24(ni). Repeating this process we construct a natural transformation between lfd14 and lfd24. Similarly, we can do this for any two extended diagrams.

The following flowchart is the algorithm for constructing a diagram morphism between two extended diagrams.

Fig. 24

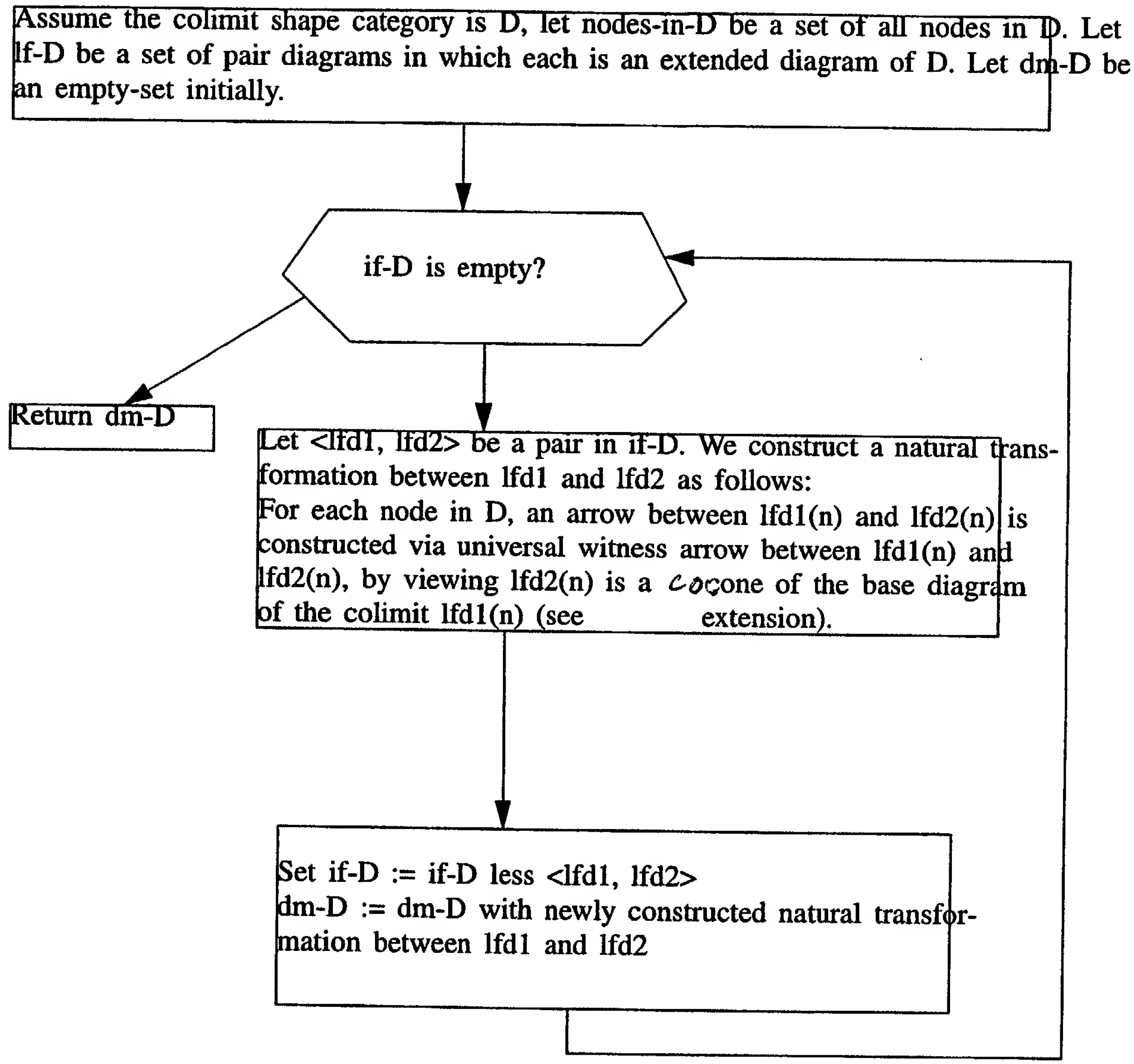

Assume the colimit shape category is D, let nodes-in-D be a set of all nodes in D. Let lf-D be a set of pair diagrams in which each is an extended diagram of D. Let dm-D be an empty-set initially.
if-D is empty?
Return dm-D
Let <lfd1, lfd2> be a pair in if-D. We construct a natural transformation between lfd1 and lfd2 as follows:
For each node in D, an arrow between lfd1(n) and lfd2(n) is constructed via universal witness arrow between lfd1(n) and lfd2(n), by viewing lfd2(n) is a cocone of the base diagram of the colimit lfd1(n) (see extension).
Set if-D := if-D less <lfd1, lfd2>
dm-D := dm-D with newly constructed natural transformation between lfd1 and lfd2

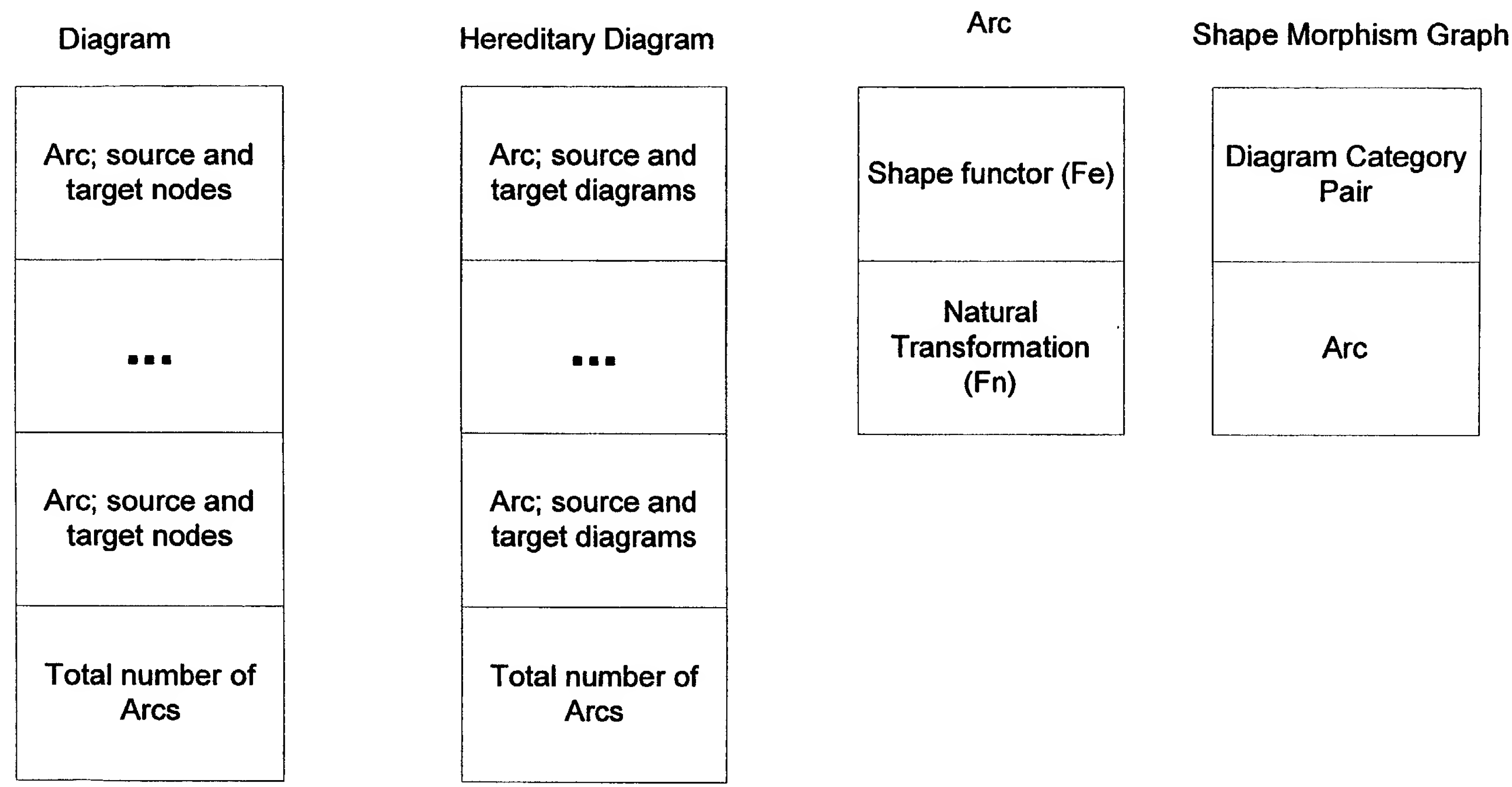
Fig. 25

The final step is to complete the colimit of the extended diagrams. The colimit is determined by computing the pointwise colimits over corresponding nodes in the extended diagrams. The morphisms are computed uniquely using universality of the pointwise colimits.

$$\begin{tikzcd}
\text{Extended diagram of d1} \arrow[r] \arrow[d] & \text{Extended diagram d2} \arrow[d] \\
\text{Extended diagram d3} \arrow[r] & \text{Colimit of Diagrams}
\end{tikzcd}$$

Taking Pointwise Colimit of Extended Diagrams (Can be done, since extended diagrams are all the same shape)

Fig. 26



Examples of Data Structures used in the Example Implementation

Fig. 27

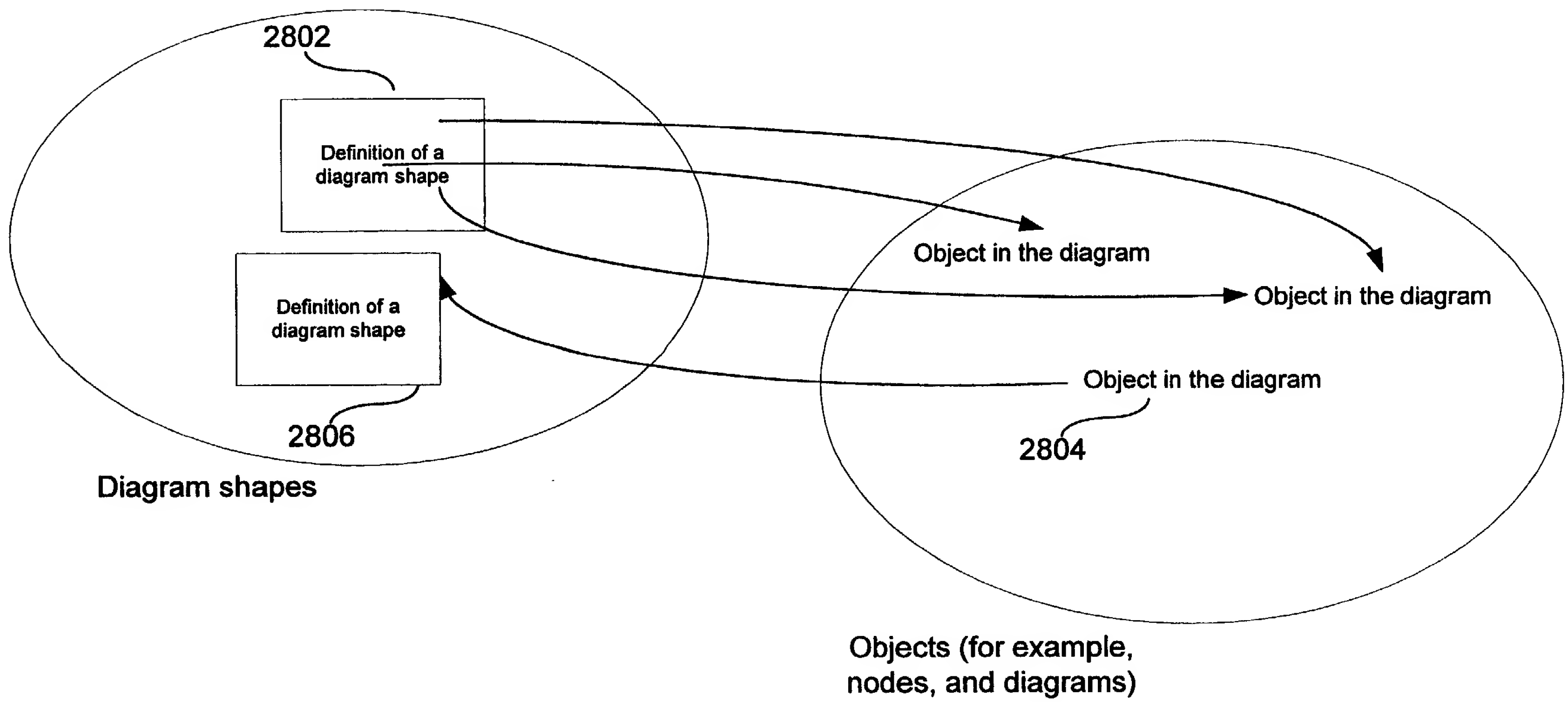

2802
Definition of a diagram shape
Definition of a diagram shape
2806
Diagram shapes
Object in the diagram
Object in the diagram
Object in the diagram
2804
Objects (for example, nodes, and diagrams)

Fig. 28

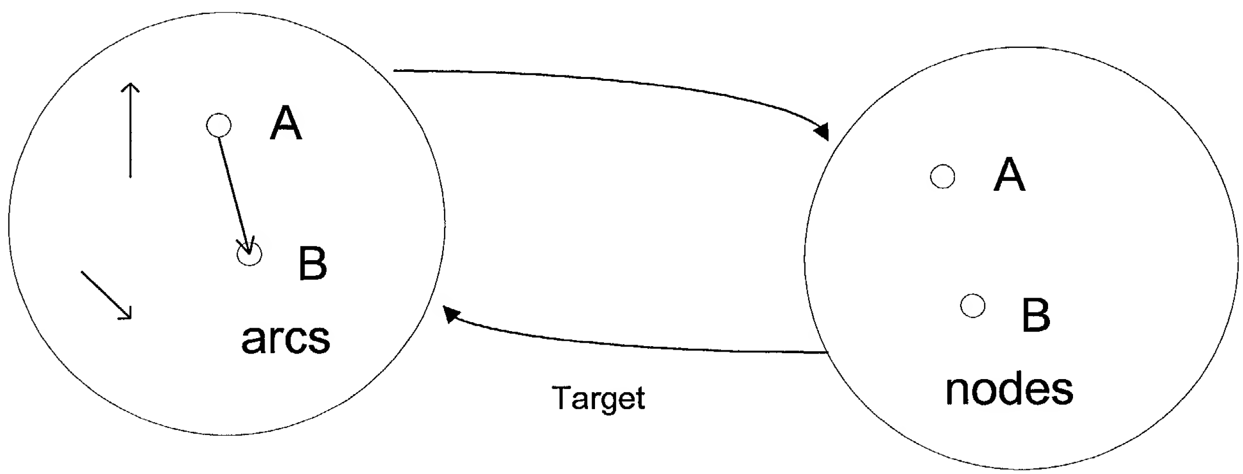


Fig. 29

0010/PTO
Rev. 6/95

U.S. Department of Commerce
Patent and Trademark Office

DECLARATION FOR UTILITY OR DESIGN PATENT APPLICATION

[X] Declaration Submitted with Initial Filing OR [ ] Declaration Submitted after Initial Filing

Attorney Docket Number	**4483 US**
First Named Inventor	**Pavlovic**
COMPLETE IF KNOWN	
Application Number	**To Be Assigned**
Filing Date	**Herewith**
Group Art Unit	**To Be Assigned**
Examiner Name	**To Be Assigned**

As a below named inventor, I hereby declare that:

My residence, post office address, and citizenship are as stated below next to my name.

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled:

METHOD AND APPARATUS FOR DETERMINING COLIMITS OF HEREDITARY DIAGRAMS

the specification of which (*Title of the Invention*)

[X] is attached hereto

OR

[ ] was filed on (MM/DD/YYYY) [____________] as United States Application Number or PCT International Application Number [__________] and was amended on (MM/DD/YYYY) [____________] (if applicable).

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment specifically referred to above.

I acknowledge the duty to disclose information which is material to patentability as defined in Title 37 Code of Federal Regulations. § 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code § 119 (a)-(d) or § 365(b) of any foreign application(s) for patent or inventor's certificate, or § 365 (a) of any PCT international application which designated at least one country other than the United States of America, listed below and have also identified below, by checking the box, any foreign application for patent or inventor's certificate, or of any PCT international application having a filing date before that of the application on which priority is claimed.

Prior Foreign Application Number(s)	Country	Foreign Filing Date (MM/DD/YYYY)	Priority Not Claimed	Certified Copy Attached? YES	NO
			[ ]	[ ]	[ ]
			[ ]	[ ]	[ ]
			[ ]	[ ]	[ ]
			[ ]	[ ]	[ ]
			[ ]	[ ]	[ ]

[ ] Additional foreign application numbers are listed on a supplemental priority sheet attached hereto:

I hereby claim the benefit under Title 35, United States Code § 119(e) of any United States provisional application(s) listed below.

Application Number(s)	Filing Date (MM/DD/YYYY)	[ ] Additional provisional application numbers are listed on a supplemental sheet attached hereto.
60/155,271	**09/19/99**	

DECLARATION	

I hereby claim the benefit under Title 35, United States Code § 120 of any United States application(s), or § 365(c) of any PCT international application designating the United States of America, listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States or PCT international application in the manner provided by the first paragraph of Title 35, United States Code § 112, I acknowledge the duty to disclose information which is material to patentability as defined in Title 37, Code of Federal Regulations § 1.56 which became available between the filing date of the prior application and the national or PCT international filing date of this application.

U.S. Parent Application Number	PCT Parent Number	Parent Filing Date (MM/DD/YYYY)	Parent Patent Number (*if applicable*)

[ ] Additional U.S. or PCT international application numbers are listed on a supplemental priority sheet attached hereto.

As a named inventor, I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and to transact all business in the Patent and Trademark Office connected therewith:

Name	Registration Number	Name	Registration Number
Greg T. Sueoka	**33,800**		
Laura A. Majerus	**33,417**		
Michelle K. Lee	**40,695**		
Tina M. Lessani	**41,150**		

[ ] Additional attorney(s) and/or agent(s) named on a supplemental sheet attached hereto.

Please direct all correspondence to:

Laura A. Majerus
Fenwick & West LLP
Two Palo Alto Square
Palo Alto, CA 94306
U.S.A.

Telephone	**(650) 858-7152**	Fax	**(650) 494-1417**

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Name of Sole or First Inventor: [ ] A petition has been filed for this unsigned inventor

Given Name	**Dusko**	Middle Initial		Family Name	**Pavlovic**	Suffix e.g. Jr.	
Inventor's Signature		Date					
Residence: City	**Palo Alto**	State	**CA**	Country	**USA**	Citizenship	**Netherlands**
Mailing Address	**1036 Colorado Avenue**						
Mailing Address							
City	**Palo Alto**	State	**CA**	Zip	**94303**	Country	**USA**

[ ] Additional inventors are being named on supplemental sheet(s) attached hereto

DECLARATION	ADDITIONAL INVENTOR(S) Supplemental Sheet

Name of Additional Joint Inventor, if any: [ ] A petition has been filed for this unsigned inventor

Given Name	**Douglas**	Middle Initial	**R.**	Family Name	**Smith**	Suffix e.g. Jr.	
Inventor's Signature					Date		
Residence: City	**Mountain View**	State	**CA**	Country	**USA**	Citizenship	**USA**
Mailing Address	**1875 Appletree Lane**						
Mailing Address							
City	**Mountain View**	State	**CA**	Zip	**94040**	Country	**USA**

Name of Additional Joint Inventor, if any: [ ] A petition has been filed for this unsigned inventor

Given Name	**Junbo**	Middle Initial		Family Name	**Liu**	Suffix e.g. Jr.	
Inventor's Signature					Date		
Residence: City	**Santa Clara**	State	**CA**	Country	**USA**	Citizenship	**China**
Mailing Address	**76 Arcadia**						
Mailing Address							
City	**Santa Clara**	State	**CA**	Zip	**95051**	Country	**USA**

Name of Additional Joint Inventor, if any: [ ] A petition has been filed for this unsigned inventor

Given Name		Middle Initial		Family Name		Suffix e.g. Jr.	
Inventor's Signature					Date		
Residence: City		State		Country		Citizenship	
Mailing Address							
Mailing Address							
City		State		Zip		Country	

Name of Additional Joint Inventor, if any: [ ] A petition has been filed for this unsigned inventor

Given Name		Middle Initial		Family Name		Suffix e.g. Jr.	
Inventor's Signature					Date		
Residence: City		State		Country		Citizenship	
Mailing Address							
Mailing Address							
City		State		Zip		Country	

[ ] Additional inventors are being named on supplemental sheet(s) attached hereto

11128/0448
3/DOCS/10
92[illegible]